aws

Benutzerhandbuch

Amazon Simple Storage Service

API-Version 2006-03-01

Amazon Simple Storage Service: Benutzerhandbuch

Table of Contents

Was ist Amazon S3?

Amazon Simple Storage Service (Amazon S3) ist ein Objektspeicherservice, der branchenführende Skalierbarkeit, Datenverfügbarkeit, Sicherheit und Leistung bietet. Kunden aller Größen und Branchen können Amazon S3 für die Speicherung und den Schutz beliebiger Datenmengen für eine Reihe von Anwendungsfällen verwenden, wie Data Lakes, Websites, mobile Anwendungen, Backup und Wiederherstellung, Archivierung, Unternehmensanwendungen, IoT-Geräte und Big-Data-Analysen. Amazon S3 bietet Verwaltungsfunktionen, mit denen Sie den Zugriff auf Ihre Daten optimieren, organisieren und konfigurieren können, um Ihre spezifischen geschäftlichen, organisatorischen und Compliance-Anforderungen zu erfüllen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Funktionen von Amazon S3

Speicherklassen

Amazon S3 bietet eine Vielzahl von Speicherklassen an, die für diverse Anwendungsfälle genutzt werden können. So können Sie beispielsweise missionskritische Produktionsdaten in S3 Standard für häufigen Zugriff speichern, Kosten sparen, indem Sie selten aufgerufene Daten in S3 Standard-

IA speichern und Daten zu den niedrigsten Kosten in S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive archivieren.

Amazon S3 Express One Zone ist eine leistungsstarke Amazon-S3-Speicherklasse mit einer einzelnen Zone, die speziell für den konsistenten Datenzugriff im einstelligen Millisekundenbereich für Ihre latenzempfindlichsten Anwendungen entwickelt wurde. S3 Express One Zone ist die heute verfügbare Cloud-Objektspeicherklasse mit der niedrigsten Latenz, mit bis zu zehnmal schnelleren Datenzugriffsgeschwindigkeiten und mit Anforderungskosten, die 50 Prozent niedriger sind als S3 Standard. S3 Express One Zone ist die erste S3-Speicherklasse, bei der Sie eine einzelne Availability Zone mit der Option auswählen können, Ihren Objektspeicher gemeinsam mit Ihren Computingressourcen zu platzieren, was die höchstmögliche Zugriffsgeschwindigkeit bietet. Um die Zugriffsgeschwindigkeit weiter zu erhöhen und Hunderttausende von Anfragen pro Sekunde zu unterstützen, werden Daten außerdem in einem neuen Bucket-Typ gespeichert: einem Amazon-S3-Verzeichnis-Bucket. Weitere Informationen finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Sie können Daten mit sich ändernden oder unbekannten Zugriffsmustern in S3 Intelligent-Tiering speichern, das die Speicherkosten optimiert, indem es Ihre Daten automatisch zwischen vier Zugriffsebenen verschiebt, wenn sich Ihre Zugriffsmuster ändern. Diese vier Zugriffsebenen umfassen zwei Zugriffsebenen mit geringer Latenz, die für häufigen und seltenen Zugriff optimiert sind, und zwei Opt-in-Archivzugriffsebenen, die für den asynchronen Zugriff auf selten abgerufene Daten ausgelegt sind.

Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen. Weitere Informationen zu S3 Glacier Flexible Retrieval finden Sie im Amazon-S3-Glacier-Entwicklerhandbuch.

Speicherverwaltung

Amazon S3 hat Speicherverwaltungsfunktionen, die Sie benutzen können, um Kosten zu verwalten, gesetzliche Anforderungen zu erfüllen, Latenz zu reduzieren und mehrere verschiedene Kopien Ihrer Daten für Compliance-Anforderungen zu speichern.

- S3-Lebenszyklus – Richten Sie eine Lebenszykluskonfiguration ein, um Ihre Objekte zu verwalten und während ihres gesamten Lebenszyklus kostengünstig zu speichern. Sie können Objekte in andere S3-Speicherklassen umstellen oder Objekte ablaufen, die das Ende ihrer Lebensdauer erreichen.
- S3-Objektsperre – Verhindern Sie, dass Amazon S3-Objekte für einen bestimmten Zeitraum oder auf unbestimmte Zeit gelöscht oder überschrieben werden. Sie können die Objektsperre

verwenden, um gesetzliche Anforderungen zu erfüllen, die write-once-read-many (WORM)-Speicher erfordern, oder um einfach eine weitere Schutzebene vor Objektänderungen und -löschungen hinzuzufügen.

- S3-Replikation – Replizieren von Objekten und deren jeweiligen Metadaten und Objekt-Tags auf einen oder mehrere Ziel-Buckets im gleichen oder anderen AWS-Regionen für reduzierte Latenz, Compliance, Sicherheit und andere Anwendungsfälle.
- S3-Batch-Vorgänge – Verwalten Sie Milliarden von Objekten skalierbar mit einer einzigen S3-API-Anforderung oder ein paar Klicks in der Amazon S3-Konsole. Sie können Batch-Vorgänge verwenden, um Vorgänge wie Kopieren, AWS-Lambda-Funktion aufrufen, und Wiederherstellen auf Millionen oder Milliarden von Objekten auszuführen.

Zugriffsverwaltung und Sicherheit

Amazon S3 bietet Funktionen für die Überwachung und Verwaltung des Zugriffs auf Ihre Buckets und Objekte. Standardmäßig werden S3-Buckets und -Objekte als privat eingestuft. Sie haben nur Zugriff auf die S3-Ressourcen, die Sie erstellen. Um detaillierte Ressourcenberechtigungen zu erteilen, die Ihren speziellen Anwendungsfall unterstützen, oder um die Berechtigungen Ihrer Amazon S3-Ressourcen zu überprüfen, können Sie die folgenden Funktionen verwenden.

- S3 öffentlichen Zugriff blockieren – Blockieren Sie den öffentlichen Zugriff auf S3-Buckets und -Objekte. Standardmäßig sind die Einstellungen für das Blockieren des öffentlichen Zugriffs auf Bucket-Ebene aktiviert. Es wird empfohlen, alle Einstellungen für das Blockieren des öffentlichen Zugriffs aktiviert zu lassen, es sei denn, Sie wissen, dass Sie eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren müssen. Weitere Informationen finden Sie unter Konfigurieren von Block-Public-Access-Einstellungen für Ihre S3-Buckets.
- AWS Identity and Access Management(IAM) – IAM ist ein Webservice, der Ihnen hilft, den Zugriff auf AWS-Ressourcen zu steuern, einschließlich Ihrer Amazon-S3-Ressourcen. Mit IAM können Sie Berechtigungen, die festlegen, auf welche AWS-Ressourcen Benutzer zugreifen dürfen, zentral verwalten. Sie verwenden IAM, um zu steuern, wer authentifiziert (angemeldet) und autorisiert (Berechtigungen besitzt) ist, Ressourcen zu nutzen.
- Bucket-Richtlinien – Verwenden Sie die IAM-basierte Richtliniensprache, um ressourcenbasierte Berechtigungen für Ihre S3-Buckets und die darin enthaltenen Objekte zu konfigurieren.
- Amazon-S3-Zugriffspunkte – Konfigurieren Sie benannte Netzwerkendpunkte mit dedizierten Zugriffsrichtlinien, um den Datenzugriff auf freigegebene Datensätze in Amazon S3 skalierbar zu verwalten.

- Zugriffskontrolllisten (ACLs) – Erteilen Sie autorisierten Benutzern Lese- und Schreibberechtigungen für einzelne Buckets und Objekte. Als allgemeine Regel sollten Sie S3-ressourcenbasierte Richtlinien (Bucket-Richtlinien und Zugriffspunkt-Richtlinien) oder IAM-Benutzerrichtlinien für die Zugriffssteuerung anstelle von ACLs verwenden. Richtlinien stellen eine vereinfachte und flexiblere Zugriffskontrolloption dar. Mit Bucket- und Zugriffspunktrichtlinien können Sie Regeln definieren, die allgemein für alle Anfragen an Ihre Amazon-S3-Ressourcen gelten. Weitere Informationen dazu, wann Sie ACLs anstelle von ressourcenbasierten Richtlinien oder IAM-Benutzerrichtlinien verwenden, finden Sie unter Richtlinien für Zugriffsrichtlinien.
- S3 Object Ownership – Übernehmen Sie die Verantwortung für jedes Objekt in Ihrem Bucket, wodurch die Zugriffsverwaltung für in Amazon S3 gespeicherte Daten vereinfacht wird. S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie ACLs deaktivieren oder aktivieren können. Standardmäßig sind ACLs deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.
- IAM-Zugriffs-Analyzer für S3 – Bewerten und überwachen Sie Ihre S3-Bucket-Zugriffsrichtlinien, um sicherzustellen, dass die Richtlinien nur den beabsichtigten Zugriff auf Ihre S3-Ressourcen zulassen.

Datenverarbeitung

Um Daten zu transformieren und Workflows auszulösen, um eine Vielzahl anderer Verarbeitungsaktivitäten maßstabsgerecht zu automatisieren, können Sie die folgenden Funktionen verwenden.

- S3 Object Lambda – Fügen Sie S3-Anforderungen GET, HEAD und LIST Ihren eigenen Code hinzu, um Daten zu ändern und zu verarbeiten, wenn sie an eine Anwendung zurückgegeben werden. Filtern Sie Zeilen, ändern Sie die Größe von Bildern dynamisch, verkleinern Sie vertrauliche Daten und vieles mehr.
- Ereignisbenachrichtigungen – Lösen Sie Workflows aus, die Amazon Simple Notification Service (Amazon SNS), Amazon Simple Queue Service (Amazon SQS) verwenden und AWS Lambda, wenn eine Änderung an Ihren S3-Ressourcen vorgenommen wird.

Speicherprotokollierung und Überwachung

Amazon S3 bietet Protokollierungs- und Überwachungstools, mit denen Sie überwachen und steuern können, wie Ihre Amazon S3-Ressourcen verwendet werden. Weitere Informationen finden Sie unter Überwachungstools.

Automatisierte Überwachungstools

- Amazon CloudWatch-Metriken für Amazon S3 – Verfolgen Sie den Betriebszustand Ihrer S3-Ressourcen und konfigurieren Sie Abrechnungswarnungen, wenn die geschätzten Gebühren einen benutzerdefinierten Schwellenwert erreichen.
- AWS CloudTrail – Aufzeichnen von Aktionen, die von einem Benutzer, einer Rolle oder einem AWS-Service in Amazon S3. CloudTrail logs durchgeführt werden, erhalten Sie eine detaillierte API-Nachverfolgung für Operationen auf S3-Bucket- und Objektebene.

Manuelle Überwachungstools

- Server-Zugriffsprotokollierung – Abrufen detaillierter Datensätze für die Anforderungen, die an einen Bucket gestellt werden. Sie können Serverzugriffsprotokolle für viele Anwendungsfälle verwenden, z. B. für die Durchführung von Sicherheits- und Zugriffsprüfungen, das Kennenlernen Ihres Kundenstamms und das Verstehen Ihrer Amazon S3-Rechnung.
- AWS Trusted Advisor – Bewerten Sie Ihr Konto mithilfe von AWS bewährte Methoden-Prüfungen, um Möglichkeiten zur Optimierung Ihrer AWS-Infrastruktur, Verbesserung der Sicherheit und Leistung, Senkung der Kosten und Überwachung von Service-Quoten zu identifizieren. Sie können dann die Empfehlungen befolgen, um Ihre Dienste und Ressourcen zu optimieren.

Analytik und Einblicke

Amazon S3 bietet Funktionen, die Ihnen dabei helfen, Einblicke in Ihre Speicherauslastung zu erhalten, sodass Sie Ihren Speicher in großem Umfang besser verstehen, analysieren und optimieren können.

- Amazon S3 Storage Lens – Verstehen, analysieren und optimieren Sie Ihren Speicher. S3 Storage Lens bietet mehr als 60 Nutzungs- und Aktivitätsmetriken und interaktive Dashboards, um Daten für Ihr gesamtes Unternehmen, bestimmte Konten, AWS-Regionen, Buckets oder Präfixe zu aggregieren.

- Speicherklassenanalys – Analysieren Sie Speicherzugriffsmuster, um zu entscheiden, wann es an der Zeit ist, Daten in eine kostengünstigere Speicherklasse zu verschieben.
- S3-Inventory mit Bestandsberichten – Überwachen und erstellen Sie Berichte zu Objekten und den entsprechenden Metadaten und konfigurieren Sie andere Amazon S3-Funktionen, um in Inventarberichten Maßnahmen zu ergreifen. Sie können beispielsweise einen Bericht über den Replikations- und Verschlüsselungsstatus Ihrer Objekte erstellen. Eine Liste aller verfügbaren Metadaten, die für jedes Objekt in Inventorberichten verfügbar sind, finden Sie unter Amazon S3-Bestandsliste.

Starke Konsistenz

Amazon S3 bietet eine starke read-after-write Konsistenz für PUT- und DELETE-Anfragen von Objekten in Ihrem Amazon S3-Bucket in allen AWS-Regionen. Dieses Verhalten gilt sowohl für Schreibvorgänge neuer Objekte als auch für PUT-Anforderungen, die vorhandene Objekte überschreiben, und DELETE-Anforderungen. Darüber hinaus sind Lesevorgänge in Amazon S3 Select, Amazon S3-Zugriffskontrolllisten (ACLs), Amazon S3-Objekt-Markierungen und Objekt-Metadaten (z. B. das HEAD-Objekt) stark konsistent. Weitere Informationen finden Sie unter Amazon S3-Datenkonsistenzmodell.

Funktionsweise von Amazon S3

Amazon S3 ist ein Objektspeicherdienst, der Daten als Objekte in Buckets speichert. Ein Objekt ist eine Datei und alle Metadaten, die diese Datei beschreiben. Ein Bucket ist ein Container für Objekte.

Um Ihre Daten in Amazon S3 zu speichern, erstellen Sie zunächst einen Bucket und geben einen Bucket-Namen und AWS-Region an. Anschließend laden Sie Ihre Daten in diesen Bucket als Objekte in Amazon S3 hoch. Jedes Objekt hat einen Schlüssel (oder Schlüsselnamen), der der eindeutige Bezeichner für das Objekt im Bucket ist.

S3 bietet Funktionen, die Sie konfigurieren können, um Ihren speziellen Anwendungsfall zu unterstützen. Beispielsweise können Sie das S3-Versioning verwenden, um mehrere Versionen eines Objekts im selben Bucket zu halten, wodurch Sie versehentlich gelöschte oder überschriebene Objekte wiederherstellen können.

Buckets und die darin enthaltenen Objekte sind privat und können nur zugegriffen werden, wenn Sie explizit Zugriffsberechtigungen erteilen. Sie können Bucket-Richtlinien,AWS Identity and Access Management-(IAM)-Richtlinien, Zugriffskontrolllisten (ACLs) und S3-Zugriffskontrolllisten für die Verwaltung des Zugriffs verwenden.

Themen

Buckets

Ein Bucket ist ein Behälter für Objekte, die in Amazon S3 gespeichert werden. Sie können beliebig viele Objekte in einem Bucket speichern und bis zu 100 Buckets in Ihrem Konto haben. Um eine Erhöhung anzufordern, rufen Sie die Service-Quotas-Konsole auf.

Jedes Objekt ist in einem Bucket enthalten. Wenn beispielsweise ein Objekt mit dem Namen `photos/puppy.jpg` im Bucket `DOC-EXAMPLE-BUCKET` in der Region USA West (Oregon) gespeichert ist, ist es über die URL `https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg` adressierbar. Weitere Informationen siehe Zugriff auf einen Bucket.

Wenn Sie einen Bucket erstellen, geben Sie einen Bucket-Namen ein und wählen die Option AWS-Region, wo sich der Bucket befindet. Nachdem Sie einen Bucket erstellt haben, können Sie den Namen des Buckets oder seiner Region nicht mehr ändern. Bucket-Namen müssen den Regeln für die Benennung von Buckets folgen. Sie können einen Bucket auch für die Verwendung des S3-Versioning oder anderer Speicherverwaltungs-Funktionen konfigurieren.

Zudem tun Buckets Folgendes:

- Sie strukturieren den Amazon S3-Namespace auf der höchsten Ebene.
- Sie identifizieren das Konto, dem die Gebühren für Datenspeicherung und -übertragung belastet werden.

- Sie stellen Zugriffssteuerungsoptionen wie Bucket-Richtlinien, Zugriffssteuerungslisten (Access Control Lists, ACLs) und S3-Zugriffskontrolllisten bereit, mit denen Sie den Zugriff auf Ihre Amazon S3-Ressourcen verwalten können.
- Sie dienen im Rahmen der Erstellung von Nutzungsberichten als Auswertungseinheit.

Weitere Informationen über Buckets finden Sie unter Bucket-Übersicht.

Objekte

Objekte sind die Grundeinheiten, die in Amazon S3 gespeichert werden. Objekte bestehen aus Objekt- und Metadaten. Metadaten bestehen aus mehreren Name/Wert-Paaren, die das Objekt beschreiben. Dazu gehören Standardmetadaten wie das Datum der letzten Aktualisierung und HTTP-Standardmetadaten wie `Content-Type`. Sie können bei der Speicherung des Objekts auch benutzerdefinierte Metadaten angeben.

Ein Objekt wird innerhalb eines Buckets eindeutig durch einen Schlüssel (Name) und eine Versions-ID identifiziert (wenn das S3-Versioning im Bucket aktiviert ist). Weitere Informationen über Objekte finden Sie unter Übersicht über Amazon-S3-Objekte.

Schlüssel

Ein Objektschlüssel (oder Schlüsselname) ist der eindeutige Bezeichner für ein Objekt in einem Bucket. Jedes Objekt in einem Bucket besitzt genau einen Schlüssel. Die Kombination aus Bucket, Objektschlüssel und optional Versions-ID (wenn das S3-Versioning für den Bucket aktiviert ist) identifiziert jedes Objekt eindeutig. Amazon S3 fungiert also als grundlegende Datenzuordnung zwischen „Bucket + Schlüssel + Version“ und dem Objekt selbst.

Jedes Objekt in Amazon S3 ist über eine Kombination von Webservice-Endpunkt, Bucket-Name, Schlüssel und wahlweise einer Version aufrufbar. So ist in der URL `https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg` `DOC-EXAMPLE-BUCKET` der Name des Buckets und `photos/puppy.jpg` der Schlüssel.

Weitere Informationen über Objektschlüssel finden Sie unter Erstellen von Objektschlüsselnamen.

S3-Versioning

Sie S3-Versioning verwenden, um mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Mit S3-Versioning können Sie jede Version jedes in Ihren Buckets gespeicherten

Objekts beibehalten, abrufen und wiederherstellen. Sie können sowohl nach unbeabsichtigten Benutzeraktionen als auch nach Anwendungsfehlern problemlos wiederherstellen.

Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Versions-ID

Wenn Sie das S3-Versioning in einem Bucket aktivieren, generiert Amazon S3 eine eindeutige Versions-ID für jedes Objekt, das dem Bucket hinzugefügt wird. Objekte, die zum Zeitpunkt der Aktivierung des Versioning bereits im Bucket vorhanden waren, haben die Versions-ID`null` . Wenn Sie diese (oder andere) Objekte mit anderen Operationen wie CopyObject und ändernPutObject, erhalten die neuen Objekte eine eindeutige Versions-ID.

Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Bucket-Richtlinie

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende AWS Identity and Access Management-(IAM)-Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt.

Bucket-Richtlinien verwenden JSON-basierte Sprache der Zugriffsrichtlinie, die standardmäßig in AWS ist. Sie können Bucket-Richtlinien verwenden, um Berechtigungen für die Objekte in einem Bucket hinzuzufügen oder zu verweigern. Bucket-Richtlinien erlauben oder verweigern Anforderungen basierend auf den Elementen in der Richtlinie, einschließlich des Anforderers, S3-Aktionen, Ressourcen und Aspekten oder Bedingungen der Anforderung (z. B. die IP-Adresse, die für die Anforderung verwendet wird). Sie können beispielsweise eine Bucket-Richtlinie erstellen, die kontoübergreifende Berechtigungen zum Hochladen von Objekten in einen S3-Bucket gewährt, während gleichzeitig sichergestellt wird, dass der Bucket-Eigentümer die volle Kontrolle über die hochgeladenen Objekte hat. Weitere Informationen finden Sie unter Beispiele für Bucket-Richtlinien.

In Ihrer Bucket-Richtlinie können Sie Platzhalterzeichen für Amazon-Ressourcennamen (ARNs) und andere Werte verwenden, um Berechtigungen für eine Teilmenge von Objekten zu erteilen. Sie können beispielsweise den Zugriff auf Gruppen von Objekten steuern, die mit einem gemeinsamen Präfixbeginnen oder mit einer bestimmten Erweiterung wie `.html` enden.

S3-Zugriffspunkte

Amazon S3-Zugriffspunkte sind benannte Netzwerkendpunkte mit dedizierten Zugriffsrichtlinien, die beschreiben, wie mit diesem Endpunkt auf Daten zugegriffen werden kann. Zugriffspunkte sind Buckets zugeordnet, mit denen Sie S3-Objektoperationen ausführen können, z. B. GetObject und PutObject. Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3.

Jeder Zugriffspunkt verfügt über eine eigene Zugriffspunktrichtlinie. Sie können Block Public Access-Einstellungen für jeden Zugriffspunkt konfigurieren. Um den Amazon S3-Datenzugriff auf ein privates Netzwerk zu beschränken, können Sie auch jeden Zugriffspunkt so konfigurieren, dass Anforderungen nur von einer Virtual Private Cloud (VPC) akzeptiert werden.

Weitere Informationen finden Sie unter Verwalten des Datenzugriffs mit Amazon S3-Zugangspunkten.

Zugriffssteuerungslisten (ACLs)

Sie können ACLs verwenden, um autorisierten Benutzern Lese- und Schreibberechtigungen für einzelne Buckets und Objekte zu erteilen. Jedem Bucket und jedem Objekt ist eine ACL als Subressource zugeordnet. Die ACL definiert, welche AWS-Konten oder -Gruppen Zugriff erhalten, und den Zugriffstyp. ACLs sind ein Zugriffssteuerungsmechanismus, der IAM vorausgeht. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Regionen

Sie können auswählen, in welcher geografischen AWS-Region Amazon S3 die erstellten Buckets speichern soll. Sie sollten eine Region im Hinblick auf Latenz, Kosten sowie Einhaltung der relevanten Vorschriften auswählen. In einer AWS-Region gespeicherte Objekte verbleiben in der Region, bis sie explizit in eine andere Region verschoben oder repliziert werden. In der Region EU (Irland) gespeicherte Objekte verlassen diese Region nicht.

> **Note**
>
> Sie können auf Amazon S3 und die zugehörigen Funktionen in AWS-Regionen zugreifen, die für das Konto aktiviert sind. Weitere Informationen darüber, wie eine Region für die Erstellung und Verwaltung von AWS-Ressourcen aktiviert wird, finden Sie unter Verwalten von AWS-Regionen in der Allgemeine AWS-Referenz.

Eine Liste der Amazon-S3-Regionen und -Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.

Amazon S3-Datenkonsistenzmodell

Amazon S3 bietet eine starke read-after-write Konsistenz für PUT- und DELETE-Anfragen von Objekten in Ihrem Amazon S3-Bucket in allen AWS-Regionen. Dieses Verhalten gilt sowohl für Schreibvorgänge neuer Objekte als auch für PUT-Anforderungen, die vorhandene Objekte überschreiben, und DELETE-Anforderungen. Darüber hinaus sind Lesevorgänge in Amazon S3 Select, Amazon S3-Zugriffskontrolllisten (ACLs), Amazon S3-Objekt-Markierungen und Objekt-Metadaten (z. B. das HEAD-Objekt) stark konsistent.

Aktualisierungen an einem einzelnen Schlüssel sind unteilbar. Wenn Sie beispielsweise eine PUT-Anforderung aus einem Thread für einen vorhandenen Schlüssel ausführen und gleichzeitig eine GET-Anforderung für denselben Schlüssel aus einem zweiten Thread ausführen, erhalten Sie entweder die alten Daten oder die neuen Daten, aber niemals teilweise neue bzw. alte oder beschädigte Daten.

Amazon S3 erzielt hohe Verfügbarkeit, indem die Daten innerhalb der AWS-Rechenzentren über mehrere Server repliziert werden. Wenn eine PUT-Anfrage erfolgreich ist, sind die Daten sicher gespeichert. Jeder Lesevorgang (GET- oder LIST-Anforderung), der nach Eingang einer

erfolgreichen PUT-Antwort eingeleitet wird, gibt die von der PUT-Anforderung geschriebenen Daten zurück. Hier finden Sie Beispiele für dieses Verhalten:

- Ein Prozess schreibt ein neues Objekt in Amazon S3 und listet sofort Schlüssel innerhalb seines Buckets auf. Das neue Objekt wird in der Liste angezeigt.
- Ein Prozess ersetzt ein vorhandenes Objekt und versucht sofort, es zu lesen. Amazon S3 gibt die neuen Daten zurück.
- Ein Prozess löscht ein vorhandenes Objekt und versucht sofort, es zu lesen. Amazon S3 gibt keine Daten zurück, da das Objekt gelöscht wurde.
- Ein Prozess löscht ein vorhandenes Objekt und listet sofort Schlüssel innerhalb seines Buckets auf. Das Objekt wird nicht im Verzeichnis angezeigt.

> **Note**
>
> - Amazon S3 unterstützt keine Objektsperre für gleichzeitige Schreibvorgänge. Wenn gleichzeitig zwei PUT-Anforderungen für denselben Schlüssel eingehen, hat die Anforderung mit dem ältesten Zeitstempel Priorität. Wenn das ein Problem ist, müssen Sie in Ihre Anwendung einen Sperrmechanismus für Objekte einbauen.
> - Updates basieren auf Schlüsseln. Es gibt keine Möglichkeit für unteilbare Aktualisierungen über Schlüssel hinweg. Sie können beispielsweise die Aktualisierung eines Schlüssels nicht von der Aktualisierung eines anderen Schlüssels abhängig machen. Dazu müssten Sie diese Funktionalität in Ihrer Anwendung implementieren.

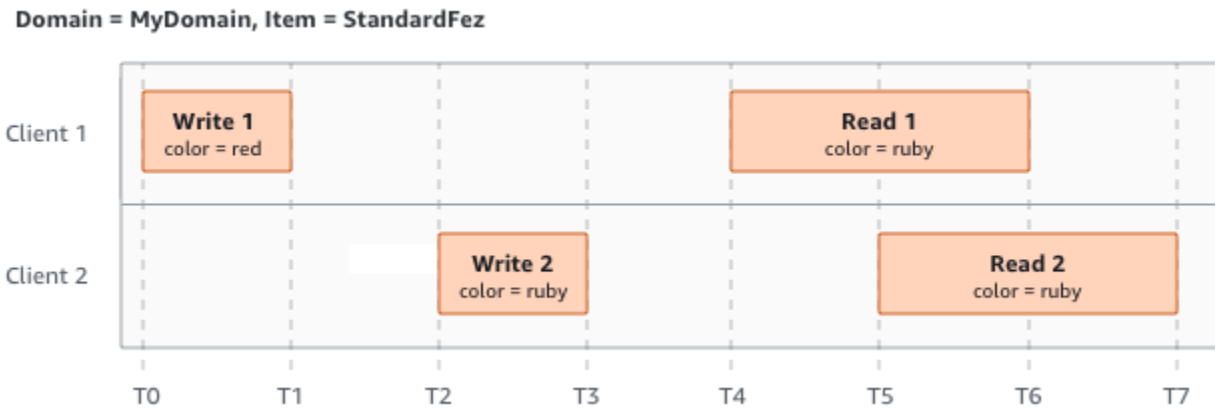
Bucket-Konfigurationen haben ein Modell mit letztendlicher Datenkonsistenz. Dies bedeutet insbesondere, dass:

- Wenn Sie einen Bucket löschen und sofort alle Buckets auflisten, wird der gelöschte Bucket möglicherweise weiterhin in der Liste angezeigt.
- Wenn Sie das Versioning für einen Bucket zum ersten Mal aktivieren, kann es einen Moment dauern, bis die Änderung vollständig verbreitet ist. Wir empfehlen, dass Sie nach dem Aktivieren des Versioning 15 Minuten warten, bevor Sie Schreibvorgänge (PUT oder DELETE-Anforderungen) für Objekte im Bucket ausführen.

Gleichzeitige Anwendungen

Dieser Abschnitt enthält Beispiele für das zu erwartende Verhalten von Amazon S3, wenn mehrere Kunden in dieselben Elemente schreiben.

In diesem Beispiel werden W1 (Lesen 1) und W2 (Lesen 2) abgeschlossen, bevor R1 (Lesen 1) und R2 (Lesen 2) starten. Da S3 stark konsistent ist, geben R1 und R2 beide `color = ruby` zurück.



Im nächsten Beispiel ist W2 vor dem Start von R1 nicht abgeschlossen. Daher könnte R1 `color = ruby` oder `color = garnet` zurückgeben. Da W1 und W2 jedoch vor dem Start von R2 fertig sind, gibt R2 zurück `color = garnet`.

Im letzten Beispiel beginnt W2, bevor W1 eine Bestätigung erhalten hat. Daher werden diese Schreibvorgänge als gleichzeitig betrachtet. Amazon S3 verwendet intern last-writer-wins Semantik, um zu bestimmen, welcher Schreibvorgang Vorrang hat. Die Reihenfolge, in der Amazon S3 die Anforderungen erhält, und die Reihenfolge, in der Anwendungen Bestätigungen erhalten, können aber aufgrund von verschiedenen Faktoren wie Netzwerklatenz nicht vorhergesagt werden. W2 könnte z. B. von einer Amazon EC2-Instance in derselben Region initiiert werden, während W1

von einem weiter entfernten Host initiiert wird. Die beste Methode, um den endgültigen Wert zu bestimmen, besteht darin, einen Lesevorgang auszuführen, nachdem beide Schreibvorgänge bestätigt wurden.

Zugehörige Services

Nachdem Sie Daten zu Amazon S3 hochgeladen haben, können Sie sie mit anderen AWS-Services nutzen. Häufig genutzte Services:

- Amazon Elastic Compute Cloud (Amazon EC2) – Bietet sichere und skalierbare Rechenkapazität in der AWS Cloud. Bei Verwendung von Amazon EC2 entfällt die Notwendigkeit von Vorabinvestitionen in Hardware. Daher können Sie Anwendungen schneller entwickeln und bereitstellen. Mit Amazon EC2 können Sie so viele oder so wenige virtuelle Server starten, wie Sie benötigen, die Sicherheit und das Netzwerk konfigurieren und den Speicher verwalten.
- Amazon EMR – Hilft Unternehmen, Forschungseinrichtungen, Datenanalysten und Entwickler einfach und kosteneffektiv riesige Datenmengen zu verarbeiten. Amazon EMR verwendet ein gehostetes Hadoop-Framework, das in der Cloud Computing-Infrastruktur von Amazon EC2 und Amazon S3 ausgeführt wird.
- AWSSnow-Familie – Hilft Kunden, die Betriebsabläufe in strengen Umgebungen ohne Rechenzentrum und an Orten ausführen müssen, an denen keine konsistente Netzwerkkonnektivität besteht. Sie können AWS-Snow-Family-Geräte verwenden, um lokal und kostengünstig auf die Speicher- und Rechenleistung des AWS Cloud an Orten zuzugreifen, an denen eine Internetverbindung möglicherweise keine Option ist.
- AWS Transfer Family – Bietet vollständig verwaltete Unterstützung für Dateiübertragungen direkt in und aus Amazon S3 oder Amazon Elastic File System (Amazon EFS) mit Secure Shell (SSH)

File Transfer Protocol (SFTP), File Transfer Protocol over SSL (FTPS) und File Transfer Protocol (FTP).

Zugriff auf Amazon S3

Sie können mit Amazon S3 auf eine der folgenden Arten arbeiten:

AWS Management Console

Die Konsole ist eine webbasierte Benutzeroberfläche für die Verwaltung von Amazon S3 und AWS-Ressourcen. Wenn Sie sich für ein AWS-Konto registriert haben, können Sie auf die Amazon S3-Konsole zugreifen, indem Sie sich bei AWS Management Console anmelden und auf der Startseite der AWS Management Console S3 auswählen.

AWS Command Line Interface

Sie können die Befehlszeilen-Tools von AWS verwenden, um Befehle in der Befehlszeile Ihres Systems auszugeben, mit denen AWS-Aufgaben (einschließlich S3) durchgeführt werden.

Das AWS Command Line Interface (AWS CLI) stellt Befehle für zahlreiche AWS-Services bereit. Die AWS CLI wird unter Windows, macOS und Linux unterstützt. Informationen zu den ersten Schritten finden Sie im AWS Command Line Interface-Benutzerhandbuch. Weitere Informationen über die Befehle für Amazon S3 finden Sie unter s3api und s3control in der AWS CLI-Befehlsreferenz.

AWS SDKs

AWS stellt SDKs (Software Development Kits) zur Verfügung, die aus Bibliotheken und Beispiel-Codes für verschiedene Programmiersprachen und Plattformen (Java, Python, Ruby, .NET, iOS, Android usw.) bestehen. Die AWS-SDKs sind gut zur Einrichtung des programmgesteuerten Zugriffs auf S3 und AWS geeignet. Amazon S3 ist ein REST-Service. Sie können Anfragen an Amazon S3 über die AWS-SDK-Bibliotheken senden, die die zugrunde liegende Amazon-S3-REST-API umschließen. Dies vereinfacht Ihre Programmieraufgaben. Beispielsweise übernehmen die SDKs Aufgaben wie das Berechnen von Signaturen, das kryptografische Signieren von Anforderungen, das Verwalten von Fehlern und das automatische erneute Ausführen von Anforderungen. Weitere Informationen über die AWS-SDKs, das Herunterladen und die Installation finden Sie unter Tools für AWS.

Jede Interaktion mit Amazon S3 erfolgt entweder authentifiziert oder anonym. Wenn Sie die AWS-SDKs verwenden, berechnen die Bibliotheken die Signatur für die Authentifizierung anhand der von

Ihnen bereitgestellten Schlüssel. Weitere Informationen darüber, wie Sie Anforderungen an Amazon S3 stellen, finden Sie unter Senden von Anforderungen.

Amazon S3-REST-API

Die Amazon S3-Architektur ist so ausgelegt, dass sie unabhängig von Programmiersprachen ist und unsere von AWS unterstützten Schnittstellen verwendet, um Objekte zu speichern und abzurufen. Sie können auf S3 und AWS programmgesteuert mithilfe der Amazon S3-REST-API zugreifen. Die REST-API ist eine HTTP-Schnittstelle zu Amazon S3. Mit REST-API verwenden Sie HTTP-Standardanfragen, um Buckets und Objekte zu erstellen, laden oder löschen.

Sie können einen beliebigen Toolkit einsetzen, der HTTP unterstützt, um die REST-API verwenden zu können. Sie können sogar einen Browser verwenden, um Objekte zu laden, wenn diese anonym lesbar sind.

Die REST-API verwendet HTTP-Standard-Header und -Statuscodes, sodass sich Standard-Browser und -Toolkits wie erwartet verhalten. In einigen Bereichen haben wir HTTP um zusätzliche Funktionen erweitert (wir haben beispielsweise Header hinzugefügt, um die Zugriffskontrolle zu unterstützen). In diesen Fällen haben wir alles dafür getan, die neue Funktion so hinzuzufügen, dass sie der standardmäßigen Nutzung von HTTP entsprechen.

Wenn Sie in Ihrer Anwendung direkte REST-API-Aufrufe senden, müssen Sie den Code für die Berechnung der Signatur schreiben und diesen der Anforderung hinzufügen. Weitere Informationen darüber, wie Sie Anforderungen an Amazon S3 stellen, finden Sie unter Senden von Anforderungen.

> **Note**
>
> Die SOAP-API-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Neuere Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST-API oder die AWS-SDKs zu verwenden.

Bezahlen für Amazon S3

Die Preise für Amazon S3 sind so ausgelegt, dass Sie nicht im Hinblick auf die Speicheranforderungen Ihrer Anwendung planen müssen. Die meisten Speicheranbieter verlangen, dass Sie eine vorgegebene Menge an Speicherkapazität und Netzwerkübertragungskapazität erwerben. Wenn Sie in diesem Szenario diese Kapazität überschreiten, wird Ihr Service abgeschaltet,

oder Sie zahlen hohe Überziehungsgebühren. Wenn Sie diese Kapazität nicht überschreiten, zahlen Sie genauso viel, als wenn Sie sie verwendet hätten.

Amazon S3 stellt Ihnen nur Gebühren für Kapazitäten in Rechnung, die Sie tatsächlich genutzt haben, ohne verborgene Kosten und Überziehungsgebühren. Dieses Modell bietet Ihnen einen Service mit variablen Kosten, der mit Ihrem Unternehmen wachsen kann und Ihnen gleichzeitig die Kostenvorteile der AWS-Infrastruktur bietet. Weitere Informationen finden Sie unter Amazon S3 – Preise.

Wenn Sie sich bei AWS registrieren, wird Ihr AWS-Konto automatisch für alle Services in AWS registriert, einschließlich Amazon S3. Es werden jedoch nur die Services in Rechnung gestellt, die Sie tatsächlich nutzen. Wenn Sie neuer Amazon S3-Kunde sind, können Sie kostenlos mit Amazon S3 beginnen. Weitere Informationen finden Sie unter Kostenloses Kontingent für AWS.

Um Ihre Rechnung anzuzeigen, navigieren Sie zu Fakturierungs- und Kostenverwaltungs-Dashboard in der AWS Billing and Cost Management-Konsole. Weitere Informationen zu AWS-Konto-Abrechnung finden Sie im AWS Billing-Benutzerhandbuch. Wenden Sie sich bei Fragen zu AWS-Abrechnungen und AWS-Konten an AWS Support.

Compliance mit PCI DSS

Amazon S3 unterstützt die Verarbeitung, Speicherung und Übertragung von Kreditkartendaten durch einen Händler oder Dienstanbieter. Außerdem wurde seine Konformität mit dem Payment Card Industry (PCI) Data Security Standard (DSS) bestätigt. Weitere Informationen über PCI DSS, einschließlich der Anforderung einer Kopie des AWS PCI Compliance Package, finden Sie unter PCI DSS Level 1.

Erste Schritte mit Amazon S3

Sie können mit Amazon S3 beginnen, indem Sie mit Buckets und Objekten arbeiten. Ein Bucket ist ein Container für Objekte. Ein Objekt ist eine Datei und alle Metadaten, die diese Datei beschreiben.

Um ein Objekt in Amazon S3 zu speichern, erstellen Sie einen Bucket und laden das Objekt dann in den Bucket hoch. Wenn sich das Objekt im Bucket befindet, können Sie es öffnen, herunterladen und verschieben. Wenn Sie kein Objekt oder einen Bucket mehr benötigen, können Sie Ihre Ressourcen aufräumen.

Mit Amazon S3 zahlen Sie nur für das, was Sie tatsächlich nutzen. Weitere Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon S3. Wenn Sie neuer Amazon-S3-Kunde sind, können Sie kostenlos mit Amazon S3 beginnen. Weitere Informationen finden Sie unter Kostenloses Kontingent für AWS.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Video: Erste Schritte mit Amazon S3

Voraussetzungen

Bevor Sie beginnen, sollten Sie sicherstellen, dass Sie die in Voraussetzung: Einrichten von Amazon S3 beschriebenen Schritte ausgeführt haben.

Themen

- Bewährte Methoden für die Zugriffssteuerung

Voraussetzung: Einrichten von Amazon S3

Wenn Sie sich bei AWS registrieren, wird Ihr AWS-Konto automatisch für alle Services in AWS registriert, einschließlich Amazon S3. Berechnet werden Ihnen aber nur die Services, die Sie nutzen.

Mit Amazon S3 zahlen Sie nur für das, was Sie tatsächlich nutzen. Weitere Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon S3. Wenn Sie neuer Amazon-S3-Kunde sind, können Sie kostenlos mit Amazon S3 beginnen. Weitere Informationen finden Sie unter Kostenloses Kontingent für AWS.

Um Amazon S3 einzurichten, führen Sie die Schritte in den folgenden Abschnitten aus.

Wenn Sie sich bei AWS registrieren und Amazon S3 einrichten, können Sie optional die Anzeigesprache in der AWS Management Console ändern. Weitere Informationen finden Sie unterChanging the language of the AWS Management Console(Ändern der Sprache der Konsole) im Handbuch Erste Schritte mit AWS Management Console.

Themen

- So melden Sie sich für ein AWS-Konto an
- Erstellen eines Administratorbenutzers

So melden Sie sich für ein AWS-Konto an

Wenn Sie kein AWS-Konto haben, führen Sie die folgenden Schritte zum Erstellen durch.

Anmeldung für ein AWS-Konto

1. Öffnen Sie https://portal.aws.amazon.com/billing/signup.
2. Folgen Sie den Online-Anweisungen.

   Bei der Anmeldung müssen Sie auch einen Telefonanruf entgegennehmen und einen Verifizierungscode über die Telefontasten eingeben.

   Wenn Sie sich für ein AWS-Konto anmelden, wird ein Root-Benutzer des AWS-Kontos erstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Als bewährte Sicherheitsmethode weisen Sie einem Administratorbenutzer Administratorzugriff zu und

verwenden Sie nur den Root-Benutzer, um Aufgaben auszuführen, die Root-Benutzerzugriff erfordern.

AWS sendet Ihnen eine Bestätigungs-E-Mail, sobald die Anmeldung abgeschlossen ist. Sie können jederzeit Ihre aktuelle Kontoaktivität anzeigen und Ihr Konto verwalten. Rufen Sie dazu https://aws.amazon.com/ auf und klicken Sie auf Mein Konto.

Erstellen eines Administratorbenutzers

Nachdem Sie sich für ein AWS-Kontoangemeldet haben, sichern Sie Ihr Root-Benutzer des AWS-Kontos, aktivieren Sie AWS IAM Identity Centerund erstellen Sie einen administrativen Benutzer, damit Sie nicht den Root-Benutzer für alltägliche Aufgaben verwenden.

Schützen Ihres Root-Benutzer des AWS-Kontos

1. Melden Sie sich bei der AWS Management Console als Kontobesitzer an, indem Sie Root-Benutzer auswählen und Ihre AWS-Konto-E-Mail-Adresse eingeben. Geben Sie auf der nächsten Seite Ihr Passwort ein.

   Hilfe bei der Anmeldung mit dem Root-Benutzer finden Sie unter Anmelden als Root-Benutzer im AWS-AnmeldungBenutzerhandbuch zu .
2. Aktivieren Sie die Multi-Faktor-Authentifizierung (MFA) für den Root-Benutzer.

   Anweisungen dazu finden Sie unter Aktivieren eines virtuellen MFA-Geräts für den Root-Benutzer Ihres AWS-Konto (Konsole) im IAM-Benutzerhandbuch.

Erstellen eines Administratorbenutzers

1. Aktivieren von IAM Identity Center.

   Anweisungen finden Sie unter Aktivieren AWS IAM Identity Center im AWS IAM Identity Center Benutzerhandbuch.
2. Im IAM Identity Center gewähren Sie einem administrativen Benutzer administrativen Zugriff.

   Ein Tutorial zur Verwendung von IAM-Identity-Center-Verzeichnis als Identitätsquelle finden Sie unter Benutzerzugriff mit dem standardmäßigen IAM-Identity-Center-Verzeichnis konfigurieren im AWS IAM Identity Center-Benutzerhandbuch.

Anmelden als Administratorbenutzer

- Um sich mit Ihrem IAM-Identity-Center-Benutzer anzumelden, verwenden Sie die Anmelde-URL, die an Ihre E-Mail-Adresse gesendet wurde, als Sie den IAM-Identity-Center-Benutzer erstellt haben.

  Hilfe bei der Anmeldung mit einem IAM-Identity-Center-Benutzer finden Sie unter Anmelden beim AWS-Zugangsportal im AWS-Anmeldung Benutzerhandbuch zu.

Schritt 1: Erstellen des ersten S3-Buckets

Nachdem Sie sich für AWS angemeldet haben, können Sie mithilfe von AWS Management Console einen Bucket in Amazon S3 erstellen. Alle Objekte in Amazon S3 werden in einem Bucket gespeichert. Bevor Sie Daten in Amazon S3 speichern können, müssen Sie einen Bucket erstellen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

> **Note**
>
> Für das Erstellen eines Buckets werden Ihnen keine Gebühren in Rechnung gestellt. Es fallen nur Gebühren für das Speichern von Objekten im Bucket und für die Übertragung von Objekten in den und aus dem Bucket an. Die Gebühren, die beim Durcharbeiten der Beispiele in diesem Handbuch anfallen, sind minimal (weniger als 1,0 USD). Weitere Information zu Speicherkosten finden Sie unter Amazon S3 – Preise.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Anschließend wird die Seite Bucket erstellen geöffnet.

4. Geben Sie unter Bucket Name (Bucket-Name) einen Namen für den Bucket ein.

   Der Bucket-Name ...:

   - Muss innerhalb einer Partition einzigartig sein. Eine Partition ist eine Gruppierung von Regionen. AWS verfügt derzeit über drei Partitionen: `aws` (Standardregionen), `aws-cn` (China-Regionen) und `aws-us-gov` (AWS GovCloud (US) Regions).
   - zwischen 3 und 63 Zeichen lang sein,
   - Darf nur aus Kleinbuchstaben, Zahlen, Punkten (.) und Bindestrichen (-) bestehen. Aus Gründen der besten Kompatibilität empfehlen wir, Punkte (.) in Bucket-Namen zu vermeiden, mit Ausnahme von Buckets, die nur für statisches Website-Hosting verwendet werden.
   - Muss mit einer Zahl oder einem Buchstaben beginnen und enden.

   Der Name eines einmal erstellter Buckets kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

   > **Important**
   >
   > Vermeiden Sie, vertrauliche Informationen, wie Kontonummern, in den Bucket-Namen einzubeziehen. Der Bucket-Name wird in der URL angezeigt, die auf die Objekte im Bucket verweist.

5. Wählen Sie in Region die AWS-Region aus, in der sich der Bucket befinden soll.

   Wählen Sie eine Region in der Nähe aus, um Latenz und Kosten gering zu halten und behördliche Vorschriften zu erfüllen. In einer Region gespeicherte Objekte verbleiben so lange in der Region, bis sie explizit in eine andere Region verschoben werden. Eine Liste der AWS-Regionen von Amazon S3 finden Sie unter AWS-Service-Endpunkte im Allgemeine Amazon Web Services-Referenz.

6. Wählen Sie unter Object Ownership eine der folgenden Einstellungen aus, um ACLs zu deaktivieren oder zu aktivieren und den Besitz von Objekten zu steuern, die in Ihren Bucket hochgeladen wurden:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Zugriffsberechtigungen für Daten im S3-Bucket. Der Bucket verwendet ausschließlich Richtlinien, um die Zugriffssteuerung zu definieren.

  Standardmäßig sind ACLs deaktiviert. Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.

  Wenn Sie die Einstellung Bucket-Eigentümer bevorzugt anwenden, damit alle Amazon-S3-Uploads die von `bucket-owner-full-control` vordefinierte ACL enthalten, können Sie eine Bucket-Richtlinie hinzufügen, die nur Objekt-Uploads zulässt, die diese ACL verwenden.
- Objekt-Writer – Das AWS-Konto, das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

> **Note**
>
> Die Standardeinstellung ist Bucket-Eigentümer erzwungen. Um die Standardeinstellung anzuwenden und ACLs deaktiviert zu lassen, ist nur die `s3:CreateBucket`-Berechtigung erforderlich. Sie müssen über die `s3:PutBucketOwnershipControls`-Berechtigung verfügen, um ACLs zu aktivieren.

7. Wählen Sie unter Einstellungen "Öffentlichen Zugriff beschränken" für diesen Bucket die Einstellungen zum Beschränken des öffentlichen Zugriffs aus, die Sie auf den Bucket anwenden möchten.

Alle vier Einstellungen zum Blockieren des öffentlichen Zugriffs sind standardmäßig aktiviert. Es wird empfohlen, alle Einstellungen aktiviert zu lassen, es sei denn, Sie wissen, dass Sie eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren müssen. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

> **Note**
>
> Zum Aktivieren aller Einstellungen zum Blockieren des öffentlichen Zugriffs ist nur die `s3:CreateBucket`-Berechtigung erforderlich. Wenn Sie eine der Einstellungen zum Blockieren des öffentlichen Zugriffs deaktivieren möchten, benötigen Sie die `s3:PutBucketPublicAccessBlock`-Berechtigung.

8. (Optional) Unter Bucket Versioning (Bucket-Versionsverwaltung) können Sie auswählen, ob Sie Varianten von Objekten in Ihrem Bucket beibehalten möchten. Weitere Informationen über das Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

   Wenn Sie die Versionsverwaltung in Ihrem Bucket deaktivieren oder aktivieren möchten, wählen Sie entweder Disable (Deaktivieren) oder Enable (Aktivieren) aus.

9. (Optional) Unter Tags können Sie auswählen, ob Sie Ihrem Bucket Tags hinzufügen möchten. Tags sind Schlüssel-Wert-Paare, die zur Kategorisierung von Speicher verwendet werden.

   Wenn Sie ein Bucket-Tag hinzuzufügen, geben Sie einen Key (Schlüssel) und optional einen Value (Wert) ein. Wählen Sie dann Add Tag (Tag hinzufügen) aus.

10. Wählen Sie unter Default encryption (Standard-Verschlüsselung) Edit (Bearbeiten) aus.

11. Wählen Sie eine der folgenden Optionen unter Verschlüsselungstyp aus, um die Standardverschlüsselung zu konfigurieren:

    - Von Amazon S3 verwalteter Schlüssel (SSE-S3)
    - AWS Key Management Service-Schlüssel (SSE-KMS)

> **Important**
>
> Wenn Sie die Option SSE-KMS für die Standardverschlüsselung verwenden, unterliegen Sie den Kontingenten der Anforderungen pro Sekunde (RPS) von AWS KMS. Weitere Informationen zu AWS KMS-Kontingenten und zum Anfordern einer

Kontingenterhöhung finden Sie unter Kontingente im Entwicklerhandbuch zu AWS Key Management Service.

Buckets und neue Objekte werden mit serverseitiger Verschlüsselung verschlüsselt. Dabei ist ein Von Amazon S3 verwalteter Schlüssel die Grundebene der Verschlüsselungskonfiguration. Weitere Informationen zur Standardverschlüsselung finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

Weitere Informationen zur Datenverschlüsselung mit der serverseitigen Amazon-S3-Verschlüsselung finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

12. Wenn Sie AWS Key Management Service-Schlüssel (SSE-KMS) ausgewählt haben, gehen Sie wie folgt vor:

    a. Geben Sie unter AWS KMS-Schlüssel Ihren KMS-Schlüssel auf eine der folgenden Arten an:

    - Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS keys wählen und anschließend den KMS-Schlüssel in der Liste der verfügbaren Schlüssel aus.

      Sowohl der Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS-Schlüssel im Entwicklerhandbuch zu AWS Key Management Service.

    - Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS key-ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.

    - Wählen Sie zum Erstellen eines neuen vom Kunden verwalteten Schlüssels in der AWS KMS-Konsole Erstellen eines KMS-Schlüssels aus.

      Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Creating Keys (Schlüssel erstellen) im AWS Key Management Service-Entwicklerhandbuch.

> **Important**
>
> Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket verfügbar sind. Die Amazon-S3-Konsole führt nur die ersten 100 KMS-Schlüssel auf, die in derselben Region wie der Bucket verfügbar sind. Wenn Sie einen KMS-Schlüssel verwenden möchten, der nicht aufgeführt ist, müssen Sie den KMS-Schlüssel-ARN eingeben. Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen und Sie müssen den KMS-Schlüssel-ARN eingeben. Weitere Informationen zu kontoübergreifenden Berechtigungen für KMS-Schlüssel finden Sie unter Erstellen von KMS-Schlüsseln, die von anderen Konten verwendet werden können im Entwicklerhandbuch zu AWS Key Management Service. Weitere Informationen zu SSE-KMS finden Sie unter Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS).
> Wenn Sie einen AWS KMS key für serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen symmetrischen KMS-Verschlüsselungsschlüssel wählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung und keine asymmetrischen KMS-Schlüssel. Weitere Informationen finden Sie unter Erkennen von symmetrischen und asymmetrischen KMS-Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service.

Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Creating Keys (Schlüssel erstellen) im AWS Key Management Service-Entwicklerhandbuch. Weitere Informationen zur Verwendung von AWS KMS mit Amazon S3 finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

b. Wenn Sie Ihren Bucket für die Verwendung der Standardverschlüsselung mit SSE-KMS konfigurieren, können Sie auch S3-Bucket-Schlüssel aktivieren. S3-Bucket-Schlüssel verringern den Anforderungsverkehr von Amazon S3 zu AWS KMS und senken die Verschlüsselungskosten. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

   Um S3-Bucket-Schlüssel zu verwenden, wählen Sie unter Bucket Key (Bucket-Schlüssel) die Option Enable (Aktivieren).

13. (Optional) Wenn Sie die S3-Objektsperre aktivieren möchten, gehen Sie wie folgt vor:

a. Wählen Sie Erweiterte Einstellungen aus.

> **Important**
>
> Durch Aktivieren der Objektsperre wird auch die Versioning für den Bucket aktiviert. Nach dem Aktivieren müssen Sie die Standardeinstellungen für die Objektsperre im Hinblick auf die (rechtliche) Aufbewahrung konfigurieren, um neue Objekte vor dem Löschen oder Überschreiben zu schützen.

b. Wenn Sie die Objektsperre aktivieren möchten, wählen Sie Enable (Aktivieren) aus, lesen Sie die angezeigte Warnung und bestätigen Sie sie.

Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

> **Note**
>
> Wenn Sie einen Bucket mit aktivierter Objektsperre erstellen möchten, benötigen Sie die folgenden Berechtigungen: `s3:CreateBucket`, `s3:PutBucketVersioning` und `s3:PutBucketObjectLockConfiguration`.

14. Wählen Sie Create Bucket (Bucket erstellen) aus.

Sie haben einen Bucket in Amazon S3 erstellt.

Nächster Schritt

Informationen zum Hinzufügen eines Objekts zu einem Bucket finden Sie unter Schritt 2: Hochladen eines Objekts in Ihren Bucket.

Schritt 2: Hochladen eines Objekts in Ihren Bucket

Nachdem Sie in Amazon S3 einen Bucket erstellt haben, können Sie ein Objekt hochladen. Ein Objekt kann eine beliebige Datei sein: eine Textdatei, ein Foto, ein Video usw.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Ein Objekt in einen Bucket hochladen

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket den Namen des Buckets aus, zu dem Sie Ihr Objekt hochladen möchten.
3. Wählen Sie im Tab Objects (Objekte) für Ihren Bucket die Option Upload (Hochladen) aus.
4. Wählen Sie unter Files and folders (Dateien und Ordner) die Option Add files (Dateien hinzufügen) aus.
5. Wählen Sie eine hochzuladende Datei und dann Öffnen aus.
6. Klicken Sie auf Hochladen.

Sie haben erfolgreich ein Objekt zu Ihrem Bucket hochgeladen.

Nächster Schritt

Informationen zum Anzeigen des Objekts finden Sie unter Schritt 3: Herunterladen eines Objekts.

Schritt 3: Herunterladen eines Objekts

Nachdem Sie ein Objekt in einen Bucket hochgeladen haben, können Sie Informationen über Ihr Objekt anzeigen und das Objekt auf Ihren lokalen Computer herunterladen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Verwenden der S3-Konsole

In diesem Abschnitt erfahren Sie, wie Sie mit der Amazon-S3-Konsole ein Objekt aus einem S3-Bucket herunterladen.

> **Note**
>
> - Sie können jeweils nur ein Objekt herunterladen.
> - Wenn Sie über die Amazon-S3-Konsole ein Objekt herunterladen, dessen Schlüsselname mit Punkt (`.`) endet, wird der Punkt aus dem Schlüsselnamen des heruntergeladenen Objekts entfernt. Wenn der Punkt am Ende des Namens des heruntergeladenen Objekts beibehalten werden soll, müssen Sie die AWS Command Line Interface (AWS CLI), die AWS-SDKs oder die Amazon-S3-REST-API verwenden.

Ein Objekt von einem S3-Bucket herunterladen

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, aus dem Sie ein Objekt herunterladen möchten.
3. Sie können ein Objekt wie folgt aus einem S3-Bucket herunterladen:
   - Aktivieren Sie das Kontrollkästchen neben dem Objekt und wählen Sie Herunterladen. Wenn Sie das Objekt in einen spezifischen Ordner herunterladen möchten, wählen Sie im Menü Aktionen die Option Herunterladen als.
   - Wenn Sie eine spezifische Version des Objekts herunterladen möchten, aktivieren Sie die Schaltfläche Versionen anzeigen (neben dem Suchfeld). Aktivieren Sie das Kontrollkästchen neben der gewünschten Version des Objekts und wählen Sie Herunterladen. Wenn Sie das Objekt in einen spezifischen Ordner herunterladen möchten, wählen Sie im Menü Aktionen die Option Herunterladen als.

Sie haben Ihr Objekt erfolgreich heruntergeladen.

Nächster Schritt

Informationen zum Kopieren und Einfügen Ihres Objekts in Amazon S3 finden Sie unter Schritt 4: Kopieren Ihres Objekts in einen Ordner.

Schritt 4: Kopieren Ihres Objekts in einen Ordner

Sie haben einem Bucket bereits ein Objekt hinzugefügt und das Objekt heruntergeladen. Jetzt erstellen Sie einen Ordner, kopieren das Objekt und fügen es in den Ordner ein.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

So kopieren Sie ein Objekt in einen Ordner

1. Wählen Sie in der Liste Buckets (Buckets) den Namen Ihres Buckets.
2. Wählen Sie Create folder (Ordner erstellen) aus und konfigurieren Sie einen neuen Ordner:
   a. Geben Sie einen Ordnernamen ein (z. B. `favorite-pics`).
   b. Als Einstellung für die Ordnerverschlüsselung wählen Sie Disable (Deaktivieren) aus.
   c. Wählen Sie Speichern.
3. Navigieren Sie zum Amazon-S3-Bucket oder -Ordner, der die Objekte enthält, die Sie kopieren möchten.
4. Markieren Sie das Kontrollkästchen links neben den Namen der Objekte, die Sie kopieren möchten.
5. Wählen Sie Actions (Aktionen) und wählen Sie aus der angezeigten Liste der Optionen die Option Copy (Kopieren) aus.

   Oder wählen Sie aus den Optionen oben rechts die Option Copy (Kopieren) aus.
6. Wählen Sie den Zielordner aus:
   a. Wählen Sie S3 durchsuchen.
   b. Wählen Sie das Optionsfeld links vom Ordnernamen aus.

Um in einen Ordner zu navigieren und einen Unterordner als Ziel auszuwählen, wählen Sie den Ordnernamen aus.

c. Wählen Sie Choose destination (Ziel wählen).

Der Pfad zu Ihrem Zielordner wird im Feld Destination (Ziel) angezeigt. In Destination (Ziel) können Sie alternativ Ihren Zielpfad eingeben, z. B. s3://*bucket-name*/*folder-name*/.

7. Wählen Sie unten rechts Copy (Kopieren).

Amazon S3 kopiert Ihre Objekte in den Zielordner.

Nächster Schritt

Informationen zum Löschen eines Objekts und eines Buckets in Amazon S3 finden Sie unter Schritt 5: Löschen Ihrer Objekte und Buckets.

Schritt 5: Löschen Ihrer Objekte und Buckets

Wenn Sie ein Objekt oder einen Bucket mehr benötigen, sollten Sie diese löschen, um weitere Gebühren zu vermeiden. Wenn Sie diesen Walkthrough für die ersten Schritte zur Übung durchgeführt haben und Ihren Bucket oder Ihre Objekte nicht verwenden möchten, sollten Sie den Bucket und die Objekte löschen, damit keine weiteren Gebühren anfallen.

Bevor Sie Ihren Bucket löschen, leeren Sie den Bucket oder löschen Sie die Objekte im Bucket. Nachdem Sie Ihre Objekte und Ihren Bucket gelöscht haben, sind die Objekte und der Bucket nicht mehr verfügbar.

Wenn Sie denselben Bucket-Namen weiter verwenden möchten, sollten Sie die Objekte löschen oder den Bucket leeren, den Bucket selbst jedoch nicht löschen. Wenn Sie einen Bucket gelöscht haben, können Sie den Namen wiederverwenden. Ein anderes AWS-Konto kann jedoch einen Bucket mit demselben Namen erstellen, bevor Sie eine Chance hatten, den Namen wiederzuverwenden.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Löschen eines Objekts

Wenn Sie nur bestimmte Objekte löschen möchten, ohne alle Objekte im Bucket zu löschen, können Sie ein Objekt löschen.

1. Wählen Sie in der Liste Buckets (Buckets) den Namen des Buckets aus, aus dem Sie ein Objekt löschen möchten.
2. Wählen Sie das zu löschende Objekt aus.
3. Wählen Sie in den Optionen oben rechts Löschen aus.
4. Geben Sie auf der Seite Objekte löschen **`delete`** ein, um das Löschen Ihrer Objekte zu bestätigen.
5. Wählen Sie Delete objects (Objekte löschen).

Leeren des Buckets

Wenn Sie Ihren Bucket löschen möchten, müssen Sie zuerst den Bucket leeren, um alle Objekte im Bucket zu löschen.

So leeren Sie einen Bucket

1. Wählen Sie in der Liste Buckets (Buckets) den Bucket aus, den Sie leeren möchten. Wählen Sie anschließend Empty (Leeren) aus.
2. Um zu bestätigen, dass Sie den Bucket leeren und alle in diesem enthaltenen Objekte löschen möchten, geben Sie unter Bucket leeren **`permanently delete`** ein.

> **Important**
>
> Das Leeren des Buckets kann nicht rückgängig gemacht werden. Dem Bucket während der Löschaktion hinzugefügte Objekte werden gelöscht.

3. Um den Bucket zu leeren und alle in diesem enthaltenen Objekte zu löschen, wählen Sie Empty (Leeren) aus.

   Anschließend wird die Seite Empty bucket: Status (Bucket leeren: Status) geöffnet, auf der Sie eine Übersicht über fehlgeschlagene und erfolgreiche Objektlöschungen anzeigen können.
4. Um zur Bucket-Liste zurückzukehren, wählen Sie Exit (Beenden) aus.

Löschen des Buckets

Nachdem Sie den Bucket geleert oder alle Objekte aus dem Bucket gelöscht haben, können Sie den Bucket löschen.

1. Um einen Bucket zu löschen, wählen Sie in der Liste Buckets (Buckets) den Bucket aus.
2. Wählen Sie Delete (Löschen).
3. Geben Sie zum Bestätigen des Löschvorgangs unter Bucket löschen den Namen des Buckets ein.

   > **Important**
   >
   > Das Löschen eines Buckets kann nicht rückgängig gemacht werden. Bucket-Namen sind eindeutig. Wenn Sie Ihren Bucket löschen, kann ein anderer AWS-Benutzer den Namen verwenden. Wenn Sie denselben Bucket-Namen weiter verwenden möchten, sollten Sie den Bucket nicht löschen. Sie sollten den Bucket stattdessen leeren und behalten.
4. Um den Bucket zu löschen, wählen Sie Delete bucket (Bucket löschen) aus.

Nächste Schritte

Im vorherigen Beispiel haben Sie gelernt, wie einige der grundlegenden Aufgaben in Amazon S3 erfolgen.

Die folgenden Themen erklären die Lernpfade, die Sie benutzen können, um sich ein vertieftes Wissen über Amazon S3 zu verschaffen, sodass Sie es in Ihren Anwendungen implementieren können.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Verstehen allgemeiner Anwendungsfälle

Sie können Amazon S3 verwenden, um Ihren speziellen Anwendungsfall zu unterstützen. Die AWS Solutions Library (Bibliothek für Lösungen) und AWS-Blog bieten anwendungsfallspezifische Informationen und Tutorials. Im Folgenden sind einige häufige Anwendungsfälle für Amazon S3 zu finden:

- Backup und Speicherung – Verwenden Sie Amazon S3 Speicherverwaltungsfunktionen, um Kosten zu verwalten, gesetzliche Anforderungen zu erfüllen, Latenz zu reduzieren und mehrere verschiedene Kopien Ihrer Daten für Compliance-Anforderungen zu speichern.
- Hosting von Anwendungen – Sie können Webanwendungen, die zuverlässig, hochgradig skalierbar und kostengünstig sind, bereitstellen, installieren und verwalten. Sie können beispielsweise Ihren Amazon-S3-Bucket so konfigurieren, dass er eine statische Website hostet. Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.
- Medienhosting – Entwicklung einer hoch verfügbaren Infrastruktur, die Uploads und Downloads von Video-, Foto- oder Musikdateien hostet.
- Softwarebereitstellung – Hosting von Softwareanwendungen, die Kunden herunterladen können.

Kontrollieren des Zugriffs auf Buckets und Objekte

Amazon S3 enthält eine Vielzahl von Sicherheitsfunktionen und -tools. Eine Übersicht finden Sie unter Bewährte Methoden für die Zugriffssteuerung.

Standardmäßig werden S3-Buckets und -Objekte als privat eingestuft. Sie haben nur Zugriff auf die S3-Ressourcen, die Sie erstellen. Sie können die folgenden Funktionen verwenden, um detaillierte Ressourcenberechtigungen zu erteilen, die Ihren speziellen Anwendungsfall unterstützen, oder um die Berechtigungen Ihrer Amazon S3-Ressourcen zu überprüfen.

- S3 öffentlichen Zugriff blockieren – Blockieren Sie den öffentlichen Zugriff auf S3-Buckets und -Objekte. Standardmäßig sind die Einstellungen für das Blockieren des öffentlichen Zugriffs auf Bucket-Ebene aktiviert.
- AWS Identity and Access Management (IAM)-Identitäten – Verwenden Sie IAM oder AWS IAM Identity Center, um IAM-Identitäten in Ihrem AWS-Konto zu erstellen, um den Zugriff auf Ihre Amazon-S3-Ressourcen zu verwalten. Sie können z. B. IAM zusammen mit Amazon S3 verwenden, um die Zugriffsart für einen Benutzer oder eine Benutzergruppe für einen Amazon-S3-Bucket zu steuern, den Ihr AWS-Konto besitzt. Weitere Informationen zu unterschiedlichen IAM-Identitäten und bewährten Verfahren finden Sie unter IAM-Identitäten (Benutzer, Benutzergruppen und Rollen) im IAM-Benutzerhandbuch.
- Bucket-Richtlinien – Verwenden Sie die IAM-basierte Richtliniensprache, um ressourcenbasierte Berechtigungen für Ihre S3-Buckets und die darin enthaltenen Objekte zu konfigurieren.
- Zugriffskontrolllisten (ACLs) – Erteilen Sie autorisierten Benutzern Lese- und Schreibberechtigungen für einzelne Buckets und Objekte. Als allgemeine Regel sollten Sie S3-ressourcenbasierte Richtlinien (Bucket-Richtlinien und Zugriffspunkt-Richtlinien) oder IAM-Benutzerrichtlinien für die Zugriffssteuerung anstelle von ACLs verwenden. Richtlinien stellen eine vereinfachte und flexiblere Zugriffskontrolloption dar. Mit Bucket- und Zugriffspunktrichtlinien können Sie Regeln definieren, die allgemein für alle Anfragen an Ihre Amazon-S3-Ressourcen gelten. Weitere Informationen dazu, wann Sie ACLs anstelle von ressourcenbasierten Richtlinien oder IAM-Benutzerrichtlinien verwenden, finden Sie unter Richtlinien für Zugriffsrichtlinien.
- S3 Object Ownership – Übernehmen Sie die Verantwortung für jedes Objekt in Ihrem Bucket, wodurch die Zugriffsverwaltung für in Amazon S3 gespeicherte Daten vereinfacht wird. S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie ACLs deaktivieren oder aktivieren können. Standardmäßig sind ACLs deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

- IAM-Zugriffs-Analyzer für S3 – Bewerten und überwachen Sie Ihre S3-Bucket-Zugriffsrichtlinien, um sicherzustellen, dass die Richtlinien nur den beabsichtigten Zugriff auf Ihre S3-Ressourcen zulassen.

Verwalten und Überwachen Ihres Speichers

- Verwalten Ihres Speichers – Nachdem Sie Buckets erstellt und Objekte in Amazon S3 hochgeladen haben, können Sie Ihren Objektspeicher verwalten. Beispielsweise können Sie die S3-Versionsverwaltung und Amazon-S3-Replikation für Notfallwiederherstellung, den S3-Lebenszyklus zur Verwaltung von Speicherkosten und die S3-Objektsperre zur Erfüllung der Compliance-Anforderungen verwenden.
- Überwachen Ihres Speichers – Überwachung ist ein wichtiger Teil, um die Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon S3 und Ihrer AWS-Lösungen aufrechtzuerhalten. Sie können Speicheraktivitäten und -kosten überwachen. Wir empfehlen auch, dass Sie von allen Teilen Ihrer AWS-Lösung Überwachungsdaten sammeln, damit Sie Ausfälle, die sich über mehrere Punkte erstrecken, leichter debuggen können.
- Analytik und Erkenntnisse – Sie können Analytik und Erkenntnisse in Amazon S3 verwenden, um Ihre Speichernutzung zu verstehen, zu analysieren und zu optimieren. Verwenden Sie z. B. Amazon S3 Storage Lens, um Ihren Speicher zu verstehen, zu analysieren und zu optimieren. S3 Storage Lens bietet mehr als 29 Nutzungs- und Aktivitätsmetriken und interaktive Dashboards, um Daten für Ihre gesamte Organisation, bestimmte Konten, Regionen, Buckets oder Präfixe zu aggregieren. Verwenden Sie Speicherklassenanalysen, um Speicherzugriffsmuster zu analysieren und zu entscheiden, wann es an der Zeit ist, Ihre Daten in eine kostengünstigere Speicherklasse zu verschieben.

Entwickeln mit Amazon S3

Amazon S3 ist ein REST-Service. Sie können Anfragen an Amazon S3 über die REST-API oder die Wrapper-Bibliotheken des AWS-SDK senden, die die zugrunde liegende Amazon S3-REST-API umschließen. Dies vereinfacht Ihre Programmier-Aufgaben. Sie können auch die AWS Command Line Interface (AWS CLI) verwenden, um Amazon S3-API-Aufrufe durchzuführen. Weitere Informationen finden Sie unter Senden von Anforderungen.

Die Amazon S3-REST-API ist eine HTTP-Schnittstelle zu Amazon S3. Mit REST API verwenden Sie HTTP-Standardanfragen, um Buckets und Objekte zu erstellen, laden oder löschen. Sie können einen beliebigen Toolkit einsetzen, der HTTP unterstützt, um die REST-API verwenden zu können.

Sie können sogar einen Browser verwenden, um Objekte zu laden, wenn diese anonym lesbar sind. Weitere Informationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der REST-API.

Damit Sie Anwendungen in der von Ihnen bevorzugten Sprache entwickeln können, stellen wir die folgenden Ressourcen zur Verfügung.

AWS CLI

Sie können auf die Funktionen von Amazon S3 mit AWS CLIzugreifen. Um die AWS CLI herunterzuladen und zu konfigurieren, siehe Entwickeln mit Amazon S3 über die AWS CLI.

Die AWS CLI bietet zwei Befehlsebenen für den Zugriff auf Amazon S3: High-Level-Befehle (s3) und Befehle der API-Ebene (s3api und s3control). Die High-Level-S3-Befehle vereinfachen das Ausführen häufiger Vorgänge. Zu diesen zählen das Erstellen, Bearbeiten und Löschen von Objekten und Buckets. Die Befehle s3api und s3control bieten direkten Zugriff auf alle Amazon S3-API-Vorgänge, mit denen Sie erweiterte Vorgänge ausführen können, die möglicherweise nicht möglich sind, wenn Sie ausschließlich High-Level-Befehle verwenden.

Eine Liste von Amazon S3-AWS CLI-Befehlen finden Sie unter s3, s3api, und s3control.

AWS-SDKs und Explorer

Sie können auch die AWS-SDKs für die Entwicklung von Anwendungen mit Amazon S3 verwenden. Die AWS-SDKs vereinfachen Ihre Programmierungs-Aufgaben, indem sie die zugrunde liegende REST-API umhüllen. Die AWS Mobile SDKs und die Amplify- JavaScript Bibliothek sind auch für die Erstellung verbundener mobiler und Webanwendungen mit verfügbarAWS.

Neben den AWS-SDKs stehen auch AWS-Explorer für Visual Studio und Eclipse für Java IDE zur Verfügung. In diesem Fall werden die SDKs und die Explorer zusammen als AWS-Toolkits gebündelt.

Weitere Informationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Beispiel-Code und Bibliotheken

Das AWS-Entwicklerzentrum und der AWS Code-Beispiel-Katalog und Bibliotheken bieten Beispiel-Code und Bibliotheken speziell für Amazon S3. Sie können mithilfe dieser Codebeispiele die Implementierung der Amazon S3-API verstehen. Sie können auch die API-Referenz zu Amazon Simple Storage Service anzeigen, um die Amazon S3-API-Vorgänge im Detail zu verstehen.

Erwerben Sie neue Kenntnisse aus Tutorials

Sie können mit step-by-step Tutorials beginnen, um mehr über Amazon S3 zu erfahren. Diese Tutorials sind für Laborumgebungen vorgesehen und sie benutzen fiktive Unternehmensnamen, Benutzernamen usw. In den Tutorials finden Sie allgemeine Anleitungen. Ohne die sorgfältige Prüfung und Anpassung an die besonderen Gegebenheiten der Umgebung Ihrer Organisation sind sie nicht zur direkten Verwendung in einer Produktionsumgebung bestimmt.

Erste Schritte

- Tutorial: Speichern und Abrufen einer Datei mit Amazon S3
- Tutorial: Erste Schritte mit S3 Intelligent-Tiering
- Tutorial: Erste Schritte mit den Amazon-S3-Glacier-Speicherklassen

Optimierung der Speicherkosten

- Tutorial: Erste Schritte mit S3 Intelligent-Tiering
- Tutorial: Erste Schritte mit den Amazon-S3-Glacier-Speicherklassen
- Tutorial: Optimieren von Kosten und Transparenz bei der Nutzung mit S3 Storage Lens

Verwalten des Speichers

- Tutorial: Erste Schritte mit Multi-Region-Zugangspunkten in Amazon S3
- Tutorial: Replizieren vorhandener Objekte in Ihren Amazon-S3-Buckets mit S3 Batch Replication

Hosten von Videos und Websites

- Tutorial: Hosten von On-Demand-Streaming-Videos mit Amazon S3 CloudFront, Amazon und Amazon Route 53
- Tutorial: Konfigurieren einer statischen Website auf Amazon S3
- Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne

Verarbeiten von Daten

- Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda
- Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend
- Tutorial: Verwenden von S3 Object Lambda, um Bilder beim Abrufen dynamisch mit Wasserzeichen zu versehen
- Tutorial: Batch-Transcodierung von Videos mit S3- AWS Lambda Batchoperationen und AWS Elemental MediaConvert

Schutz von Daten

- Tutorial: Überprüfen der Integrität von Daten in Amazon S3 mit zusätzlichen Prüfsummen
- Tutorial: Replizieren von Daten innerhalb und zwischen AWS-Regionen mit S3 Replication
- Tutorial: Schutz von Daten in Amazon S3 vor dem versehentlichen Löschen oder Anwendungsfehlern mithilfe von S3 Versioning, S3 Object Lock und S3 Replication
- Tutorial: Replizieren vorhandener Objekte in Ihren Amazon S3-Buckets mit S3 Batch Replication

Lernen Sie Trainings und Support kennen

Sie können von AWS-Experten lernen, um Ihre Fähigkeiten zu fördern und kompetente Unterstützung bei der Erreichung Ihrer Ziele zu erhalten.

- Training – Trainingsressourcen bieten einen praktischen Ansatz zum Erlernen von Amazon S3. Weitere Informationen finden Sie unter AWS-Schulungen und -Zertifizierung und AWS-Online-Kundendienstmitarbeiter.
- Diskussionsforen – Im Forum können Sie Beiträge überprüfen, um zu erfahren, was mit Amazon S3 alles möglich ist und was nicht. Sie können auch Ihre Fragen posten. Weitere Informationen finden Sie unter Diskussionsforen.
- Technischer Support – Wenn Sie weitere Fragen haben, können Sie sich an den Technischen Support wenden.

Bewährte Methoden für die Zugriffssteuerung

Amazon S3 enthält eine Vielzahl von Sicherheitsfunktionen und -tools. Die folgenden Szenarien dienen als Leitfaden dafür, welche Tools und Einstellungen Sie bei der Ausführung bestimmter Aufgaben oder in bestimmten Umgebungen verwenden sollten. Die ordnungsgemäße Anwendung dieser Tools kann dazu beitragen, die Integrität Ihrer Daten zu wahren und sicherzustellen, dass die vorgesehenen Benutzer auf die Ressourcen zugreifen können.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Erstellen eines neuen Buckets

Wenn Sie einen neuen Bucket erstellen, sollten Sie die folgenden Tools und Einstellungen anwenden, um sicherzustellen, dass Ihre Amazon S3-Ressourcen geschützt sind.

S3 Object Ownership zur Vereinfachung der Zugriffskontrolle

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch Zugriffssteuerungslisten (ACLs) deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer jedes Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Richtlinien.

Object Ownership verfügt über drei Einstellungen, mit denen Sie sowohl die Eigentümerschaft von Objekten, die in Ihren Bucket hochgeladen werden, steuern als auch ACLs deaktivieren oder aktivieren können:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Berechtigungen für Daten im S3-Bucket. Der Bucket verwendet ausschließlich Richtlinien, um die Zugriffssteuerung zu definieren.

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.
- Objekt-Writer – Das AWS-Konto, das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Block Public Access

S3 Block Public Access stellt vier Einstellungen bereit, die Ihnen helfen, eine versehentliche Offenlegung Ihrer S3-Ressourcen zu vermeiden. Sie können diese Einstellungen in beliebiger Kombination auf einzelne Zugriffspunkte, Buckets oder auf ganze AWS-Konten anwenden. Wenn Sie eine Einstellung auf ein Konto anwenden, gilt sie für alle Buckets und Zugriffspunkte, die dem Konto gehören. Die vier Einstellungen Blockieren jeglichen öffentlichen Zugriffs sind für neue Buckets standardmäßig aktiviert. Es wird empfohlen, alle vier Einstellungen aktiviert zu lassen, es sei denn, Sie wissen, dass Sie eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren müssen. Sie können einzelne Einstellungen zum Blockieren des öffentlichen Zugriffs mithilfe der Amazon-S3-Konsole ändern.

Weitere Informationen finden Sie unter Die Bedeutung von „öffentlich“.

Wenn Sie bestimmten Benutzern Zugriff gewähren möchten, empfehlen wir, AWS Identity and Access Management (IAM)-Richtlinien zu verwenden, statt alle Einstellungen zum Blockieren des öffentlichen Zugriffs zu deaktivieren. Es ist eine bewährte Sicherheitsmethode, das Blockieren

des öffentlichen Zugriffs zu aktivieren. Durch die Verwendung des Blockierens des öffentlichen Zugriffs mit IAM-Identitäten wird sichergestellt, dass eine von einer Einstellung zum Blockieren des öffentlichen Zugriffs blockierte Operation zurückgewiesen wird, es sei denn, dem angeforderten Benutzer wurde eine spezifische Berechtigung erteilt.

Weitere Informationen finden Sie unter Block Public Access-Einstellungen.

Erteilung von Zugriff mit IAM-Identitäten

Verwenden Sie beim Einrichten von Konten für neue Teammitglieder, die S3-Zugriff benötigen, IAM-Benutzer und -Rollen, um nur die mindestens erforderlichen Berechtigungen zu erteilen. Sie können auch eine Form der IAM-Multi-Factor Authentication (MFA) implementieren, um eine robuste Identitätsgrundlage zu unterstützen. Mithilfe von IAM-Identitäten können Sie Benutzern spezifische Berechtigungen erteilen und angeben, auf welche Ressourcen sie zugreifen und welche Aktionen sie ausführen können. IAM-Identitäten stellen fortschrittliche Funktionen bereit, einschließlich der Möglichkeit, Benutzer zur Eingabe von Anmeldeinformationen zu verpflichten, bevor sie auf freigegebene Ressourcen zugreifen, und Berechtigungshierarchien auf verschiedene Objekte in einem einzelnen Bucket anzuwenden.

Weitere Informationen finden Sie unter Beispiel 1: Bucket-Eigentümer erteilt seinen Benutzern Bucket-Berechtigungen.

Bucket-Richtlinien

Mit Bucket-Richtlinien können Sie den Bucket-Zugriff personalisieren, um sicherzustellen, dass nur die Benutzer, die Sie genehmigt haben, auf Ressourcen zugreifen und Aktionen in ihnen ausführen können. Es wird empfohlen, zusätzlich zu den Bucket-Richtlinien die Einstellungen zum Blockieren des öffentlichen Zugriffs auf Bucket-Ebene zu verwenden, um den öffentlichen Zugriff auf Ihre Daten weiter einzuschränken.

Weitere Informationen finden Sie unter Verwenden von Bucket-Richtlinien.

Vermeiden Sie beim Erstellen von Richtlinien die Verwendung von Platzhalterzeichen (*) im `Principal`-Element, da ein Platzhalterzeichen allen Benutzer den Zugriff auf Ihre Amazon-S3-Ressourcen ermöglicht. Listen Sie stattdessen die Benutzer oder Gruppen explizit auf, die auf den Bucket zugreifen dürfen. Anstatt ein Platzhalterzeichen für ihre Aktionen einzufügen, gewähren Sie ihnen spezifische Berechtigungen, wenn zutreffend.

Zur Unterstützung der Praxis der geringsten Berechtigungen sollten `Deny`-Anweisungen im `Effect`-Element so umfassend wie möglich und `Allow`-Anweisungen so eng wie möglich

verfasst werden. `Deny`-Effekte in Verbindung mit der `s3:*`-Aktion sind eine weitere gute Möglichkeit, bewährte Methoden für die Anmeldung der Benutzer zu implementieren, die in Richtlinienbedingungsanweisungen enthalten sind.

Weitere Informationen zur Angabe von Bedingungen für das Inkrafttreten einer Richtlinie finden Sie unter Beispiele für Amazon-S3-Bedingungsschlüssel.

Buckets in einer VPC-Umgebung

Wenn Sie Benutzer in einer Unternehmensumgebung hinzufügen, können Sie einen Virtual Private Cloud (VPC)-Endpunkt verwenden, um Benutzern in Ihrem virtuellen Netzwerk den Zugriff auf Ihre Amazon S3-Ressourcen zu ermöglichen. VPC-Endpunkte ermöglichen Entwicklern die Bereitstellung spezifischer Zugriffe und Berechtigungen für Benutzergruppen basierend auf dem Netzwerk, mit dem die jeweiligen Benutzer verbunden sind. Anstatt jeden einzelnen Benutzer einer IAM-Rolle oder -Gruppe hinzuzufügen, können Sie VPC-Endpunkte verwenden, um den Bucket-Zugriff abzulehnen, wenn die Anforderung nicht vom angegebenen Endpunkt stammt.

Weitere Informationen finden Sie unter Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien.

Speichern und Freigeben von Daten

Verwenden Sie die folgenden Tools und bewährten Methoden, um Ihre Amazon S3-Daten zu speichern und freizugeben.

Versioning und Objektsperre für Datenintegrität

Wenn Sie die Amazon-S3-Konsole zum Verwalten von Buckets und Objekten verwenden, sollten Sie S3 Versioning und S3-Objektsperre implementieren. Diese Funktionen verhindern versehentliche Änderungen kritischer Daten und ermöglichen Ihnen das Rollback unbeabsichtigter Aktionen. Diese Rollback-Funktion ist besonders nützlich, wenn mehrere Benutzer mit vollständigen Schreib- und Ausführungsberechtigungen auf die Amazon-S3-Konsole zugreifen.

Weitere Informationen über S3 Versioning finden Sie unter Verwenden der Versioning in S3-Buckets. Weitere Informationen über die Objektsperre finden Sie unter Verwenden der S3-Objektsperre.

Verwaltung des Objektlebenszyklus hinsichtlich Kosteneffizienz

Um Objekte so zu verwalten, dass sie während des gesamten Lebenszyklus kostengünstig gespeichert werden, können Sie Lebenszykluskonfigurationen mit der S3-Versionsverwaltung

verbinden. Lebenszykluskonfigurationen definieren Aktionen, die Amazon S3 während des Lebenszyklus eines Objekts ausführen soll. Sie können beispielsweise eine Lebenszykluskonfiguration erstellen, die Objekte nach einem bestimmten Zeitraum in eine andere Speicherklasse überführt, archiviert oder löscht. Sie können eine Lebenszykluskonfiguration für alle Objekte im Bucket oder für eine Teilmenge von Objekten definieren, indem Sie ein gemeinsames Präfix oder Tag verwenden.

Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Regionsübergreifende Replikation für mehrere Standorte

Wenn Sie Buckets erstellen, auf die von verschiedenen Standorten aus zugegriffen wird, erwägen Sie die Implementierung einer regionsübergreifenden S3-Replikation. Die regionsübergreifende Replikation trägt dazu bei, dass alle Benutzer Zugriff auf die Ressourcen erhalten, die sie benötigen, und erhöht die betriebliche Effizienz. Die regionsübergreifende Replikation verbessert die Verfügbarkeit durch das Kopieren von Objekten über S3-Buckets über verschiedene AWS-Regionen hinweg. Die Verwendung dieser Funktion erhöht jedoch die Speicherkosten.

Weitere Informationen finden Sie unter Replizieren von Objekten.

Berechtigungen für ein sicheres statisches Website-Hosting

Wenn Sie einen Bucket konfigurieren, der als eine öffentlich zugängliche statische Website verwendet werden soll, müssen Sie alle Block-Public-Access-Einstellungen deaktivieren. Wenn Sie die Bucket-Richtlinie für Ihre statische Website schreiben, achten Sie darauf, nur `s3:GetObject`-Aktionen und keine `ListObject`- oder `PutObject`-Berechtigungen zuzulassen. Damit wird sichergestellt, dass Benutzer nicht alle Objekte im Bucket anzeigen oder eigene Inhalte hinzufügen können.

Weitere Informationen finden Sie unter Festlegen von Berechtigungen für den Website-Zugriff.

Sie sollten alle Einstellungen für das Blockieren des öffentlichen Zugriffs aktiviert lassen. Wenn Sie alle vier Block Public Access-Einstellungen aktiviert lassen und eine statische Website hosten möchten, können Sie die Amazon- CloudFront Ursprungszugriffssteuerung (OAC) verwenden. Amazon CloudFront bietet die Funktionen, die zum Einrichten einer sicheren statischen Website erforderlich sind. Statische Amazon S3-Websites unterstützen nur HTTP-Endpunkte. CloudFront verwendet den dauerhaften Speicher von Amazon S3 und bietet gleichzeitig zusätzliche Sicherheitsheader wie HTTPS. HTTPS erhöht die Sicherheit, indem eine normale HTTP-Anforderung verschlüsselt und vor gängigen Cyberangriffen geschützt wird.

Weitere Informationen finden Sie unter Erste Schritte mit einer sicheren statischen Website im Amazon- CloudFront Entwicklerhandbuch.

Freigabe von Ressourcen

Es gibt verschiedene Möglichkeiten, wie Sie Ressourcen für eine bestimmte Benutzergruppe freigeben können. Sie können die folgenden Tools verwenden, um eine Gruppe von Dokumenten oder anderen Ressourcen für eine einzelne Gruppe von Benutzern, eine Abteilung oder einen Standort freizugeben. Obwohl diese Tools zum Erreichen des gleichen Ziels verwendet werden können, sind einige Tools möglicherweise besser als andere für die vorhandenen Einstellungen geeignet.

S3 Object Ownership

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch Zugriffssteuerungslisten (ACLs) deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer jedes Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Richtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen.

Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Benutzerrichtlinien

Mittels IAM-Gruppen und Benutzerrichtlinien können Sie Ressourcen für eine begrenzte Gruppe von Personen freigeben. Wenn Sie neue IAM-Benutzer erstellen, werden Sie aufgefordert, sie zu erstellen und einer Gruppe hinzuzufügen. Sie können Benutzer jedoch jederzeit erstellen und Gruppen hinzufügen. Wenn die Personen, für die Sie diese Ressourcen freigeben möchten, bereits in IAM eingerichtet sind, können Sie sie einer gemeinsamen Gruppe hinzufügen. Anschließend können Sie eine IAM-Benutzerrichtlinie verwenden, um den Bucket für diese Gruppe freizugeben. Sie können IAM-Benutzerrichtlinien auch verwenden, um einzelne Objekte innerhalb eines Buckets freizugeben.

Weitere Informationen finden Sie unter Einem IAM-Benutzer den Zugriff auf einen Ihrer Buckets erlauben.

Zugriffskontrolllisten

In der Regel empfehlen wir, dass Sie S3-Bucket-Richtlinien oder IAM-Benutzerrichtlinien für die Zugriffskontrolle verwenden. Die Verwendung von Richtlinien anstelle von ACLs vereinfacht die Berechtigungsverwaltung. Amazon-S3-ACLs sind der ursprüngliche Zugriffskontrollmechanismus in Amazon S3, der älter als IAM ist. Bestimmte Zugriffskontrollszenarien erfordern jedoch die Verwendung von ACLs. Angenommen, ein Bucket-Eigentümer möchte Berechtigungen für den Zugriff auf Objekte erteilen, aber nicht alle Objekte im Bucket gehören dem Bucket-Eigentümer. In diesem Fall muss der Objekteigentümer zuerst dem Bucket-Eigentümer über eine Objekt-ACL Berechtigungen erteilen.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Mit Object Ownership können Sie ACLs deaktivieren und sich auf Richtlinien für die Zugriffssteuerung verlassen. Wenn Sie ACLs deaktivieren, können Sie einfach einen Bucket mit Objekten verwalten, die von verschiedenen AWS-Konten hochgeladen wurden. Sie als Bucket-Eigentümer besitzen alle Objekte im Bucket und können den Zugriff darauf mithilfe von Richtlinien verwalten.

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien verwenden, um Zugriff auf Ihren Bucket und die darin enthaltenen Objekte zu gewähren. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Weitere Informationen zur Verwendung von ACLs finden Sie unter Beispiel 3: Bucket-Eigentümer, der Berechtigungen für Objekte erteilt, die ihm nicht gehören.

Präfixe

Wenn bestimmte Ressourcen aus einem Bucket freigeben möchten, können Sie mithilfe von Präfixen Berechtigungen auf Ordnerebene replizieren. Die Amazon-S3-Konsole unterstützt das Ordnerkonzept als Möglichkeit zum Gruppieren von Objekten mithilfe eines freigegebenen Namenspräfixes für

Objekte. Wenn Sie einem IAM-Benutzer anschließend explizite Berechtigungen für den Zugriff auf die Ressourcen erteilen möchten, die diesem Präfix zugeordnet sind, können Sie das Präfix innerhalb der Bedingungen der IAM-Benutzerrichtlinie eines Benutzers angeben.

Weitere Informationen finden Sie unter Organisieren von Objekten in der Amazon S3-Konsole mithilfe von Ordnern.

Tagging

Wenn Sie für die Speicherkategorisierung Objektmarkierung verwenden, können Sie Objekte, die mit einem bestimmten Wert markiert wurden, für bestimmte Benutzer freigeben. Mithilfe der Ressourcenmarkierung können Sie den Zugriff auf Objekte basierend auf den Markierungen steuern, die mit der Ressource verknüpft sind, auf die ein Benutzer zugreifen möchte. Verwenden Sie die Bedingung `ResourceTag/key-name` innerhalb einer IAM-Benutzerrichtlinie, um den Zugriff auf die markierten Ressourcen zu ermöglichen.

Weitere Informationen finden Sie unter Steuern des Zugriffs auf AWS-Ressourcen mit Ressourcen-Tags im IAM-Benutzerhandbuch.

Schutz von Daten

Mit den folgenden Tools können Sie Daten während der Übertragung und im Ruhezustand schützen. Beides ist für die Wahrung der Integrität und Zugänglichkeit Ihrer Daten von kritischer Bedeutung.

Objekt-Verschlüsselung

Amazon S3 bietet verschiedene Optionen für die Objektverschlüsselung, um Daten während der Übertragung und im Ruhezustand zu schützen. Bei der serverseitigen Verschlüsselung wird das Objekt verschlüsselt, bevor es auf Datenträgern im Rechenzentrum gespeichert wird. Wenn die Objekte heruntergeladen werden, werden sie wieder entschlüsselt. Wenn Sie Ihre Anforderung authentifizieren und Zugriffsberechtigungen besitzen, gibt es in Bezug auf die Art und Weise, wie Sie auf verschlüsselte oder nicht verschlüsselte Objekte zugreifen, keinen Unterschied. Beim Einrichten der serverseitigen Verschlüsselung können Sie aus drei sich gegenseitig ausschließende Optionen wählen:

- Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)
- Serverseitige Verschlüsselung mit Schlüsseln, die von AWS Key Management Service (AWS KMS) (SSE-KMS) verwaltet werden
- Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Weitere Informationen finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung.

Clientseitige Verschlüsselung ist der Vorgang, bei dem Daten verschlüsselt werden, bevor sie an Amazon S3 gesendet werden. Weitere Informationen finden Sie unter Schützen von Daten mithilfe der clientseitigen Verschlüsselung.

Signaturmethoden

Signature Version 4 ist der Prozess für die Hinzufügung von Authentifizierungs-Informationen zu AWS-Anforderungen, die über HTTP gesendet werden. Aus Sicherheitsgründen müssen die meisten Anforderungen an AWS mit einem Zugriffsschlüssel signiert werden, der aus einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel besteht. Diese beiden Schlüssel werden in der Regel als Sicherheitsanmeldeinformationen bezeichnet.

Weitere Informationen finden Sie unter Authenticating Requests (Authentifizierung von Anforderungen) (AWS Signature Version 4) und Signature Version 4 signing process (Signaturprozess).

Protokollierung und Überwachung

Die Überwachung ist wichtig, um die Zuverlässigkeit, Verfügbarkeit und Leistung Ihrer Amazon S3-Lösungen zu wahren. Dank der Überwachung können Sie einen Multi-Punkt-Fehler leicht debuggen, wenn ein solcher Fehler auftritt. Die Protokollierung kann einen Einblick in alle Fehler geben, die Benutzer erhalten und wann welche Anfragen gestellt werden. AWS bietet mehrere Tools zur Überwachung Ihrer Amazon S3-Ressourcen:

- Amazon CloudWatch
- AWS CloudTrail
- Amazon S3-Zugriffsprotokolle
- AWS Trusted Advisor

Weitere Informationen finden Sie unter Protokollierung und Überwachung in Amazon S3.

Amazon S3 ist in AWS CloudTrail integriert. Dieser Service zeichnet die Aktionen eines Benutzers, einer Rolle oder eines AWS-Service in Amazon S3 auf. Diese Funktion kann mit Amazon kombiniert werden GuardDuty, das Bedrohungen für Ihre Amazon S3-Ressourcen überwacht, indem CloudTrail Verwaltungsereignisse und CloudTrail S3-Datenereignisse analysiert werden. Diese Datenquellen überwachen verschiedene Arten von Aktivitäten. Zu Amazon S3-bezogenen CloudTrail Verwaltungsereignissen gehören beispielsweise Vorgänge, die S3-Projekte auflisten oder

konfigurieren. GuardDuty analysiert S3-Datenereignisse aus all Ihren S3-Buckets und überwacht sie auf böswillige und verdächtige Aktivitäten.

Weitere Informationen finden Sie unter Amazon S3-Schutz in Amazon GuardDuty im Amazon-GuardDuty Benutzerhandbuch.

Tutorials

In den folgenden Tutorials werden vollständige end-to-end Verfahren für allgemeine Amazon S3-Aufgaben vorgestellt. Diese Tutorials sind für Laborumgebungen vorgesehen und sie benutzen fiktive Unternehmensnamen, Benutzernamen usw. In den Tutorials finden Sie allgemeine Anleitungen. Ohne die sorgfältige Prüfung und Anpassung an die besonderen Gegebenheiten der Umgebung Ihrer Organisation sind sie nicht zur direkten Verwendung in einer Produktionsumgebung bestimmt.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Erste Schritte

- Tutorial: Speichern und Abrufen einer Datei mit Amazon S3
- Tutorial: Erste Schritte mit S3 Intelligent-Tiering
- Tutorial: Erste Schritte mit den Amazon-S3-Glacier-Speicherklassen

Optimierung der Speicherkosten

- Tutorial: Erste Schritte mit S3 Intelligent-Tiering
- Tutorial: Erste Schritte mit den Amazon-S3-Glacier-Speicherklassen
- Tutorial: Optimieren von Kosten und Transparenz bei der Nutzung mit S3 Storage Lens

Verwalten des Speichers

- Tutorial: Erste Schritte mit Multi-Region-Zugangspunkten in Amazon S3
- Tutorial: Replizieren vorhandener Objekte in Ihren Amazon-S3-Buckets mit S3 Batch Replication

Hosten von Videos und Websites

- Tutorial: Hosten von On-Demand-Streaming-Videos mit Amazon S3 CloudFront, Amazon und Amazon Route 53
- Tutorial: Konfigurieren einer statischen Website auf Amazon S3
- Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne

Verarbeiten von Daten

- Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda
- Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend
- Tutorial: Verwenden von S3 Object Lambda, um Bilder beim Abrufen dynamisch mit Wasserzeichen zu versehen
- Tutorial: Batch-Transcodierung von Videos mit S3- AWS Lambda Batchoperationen und AWS Elemental MediaConvert

Schutz von Daten

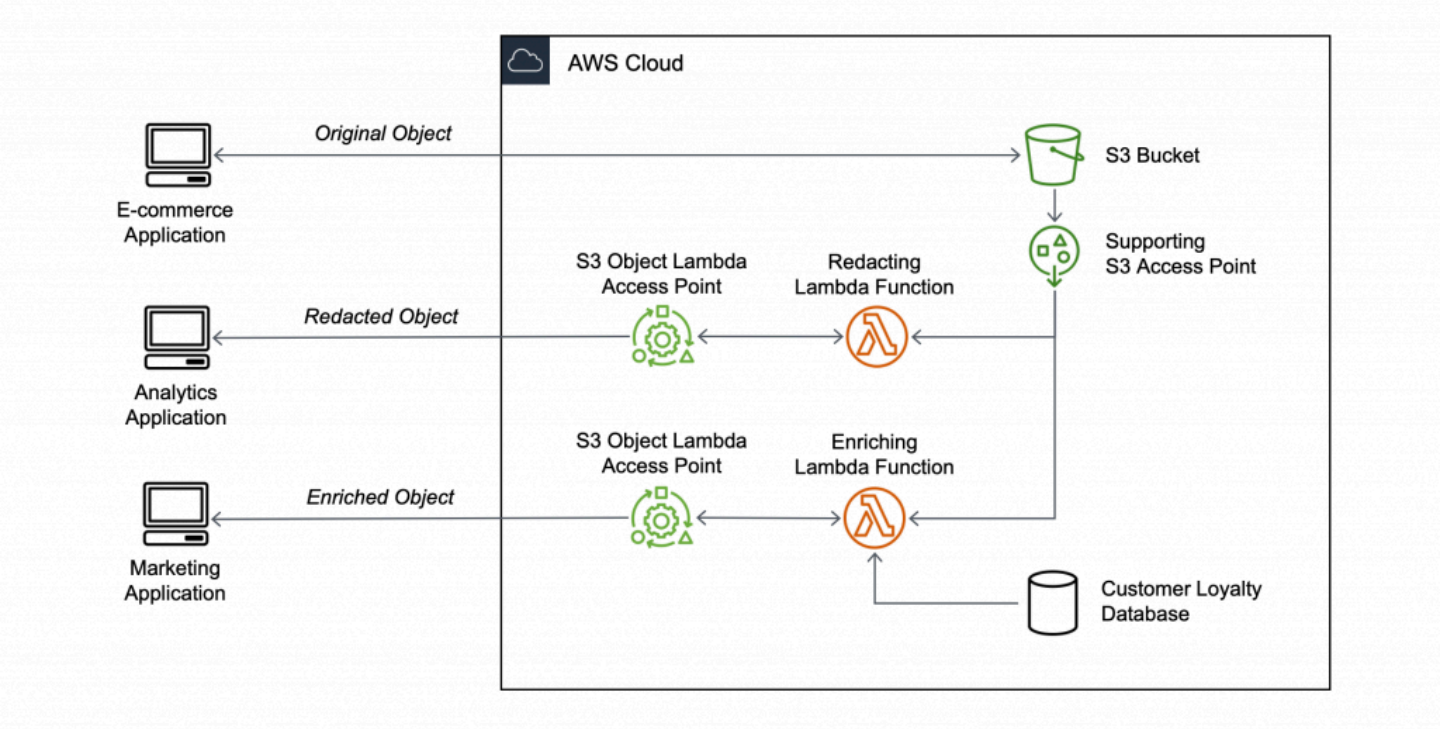
- Tutorial: Überprüfen der Integrität von Daten in Amazon S3 mit zusätzlichen Prüfsummen
- Tutorial: Replizieren von Daten innerhalb und zwischen AWS-Regionen mithilfe von S3 Replication
- Tutorial: Schutz von Daten in Amazon S3 vor dem versehentlichen Löschen oder Anwendungsfehlern mithilfe von S3 Versioning, S3 Object Lock und S3 Replication
- Tutorial: Replizieren vorhandener Objekte in Ihren Amazon-S3-Buckets mit S3 Batch Replication

Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda

Wenn Sie Daten in Amazon S3 speichern, können Sie sie für mehrere Anwendungen verwenden. Jede Anwendung hat jedoch möglicherweise eindeutige Anforderungen an das Datenformat und benötigt möglicherweise eine Änderung oder Verarbeitung Ihrer Daten für einen bestimmten Anwendungsfall. Beispielsweise kann ein von einer E-Commerce-Anwendung erstellter Datensatz persönlich identifizierbare Informationen (PII) enthalten. Wenn dieselben Daten für Analytik

verarbeitet werden, wird diese PII nicht benötigt und sollte redigiert werden. Wenn jedoch derselbe Datensatz für eine Marketingkampagne verwendet wird, müssen Sie die Daten möglicherweise mit zusätzlichen Details wie Informationen aus der Kundenbindungsdatenbank anreichern.

Mit S3 Object Lambda können Sie Ihren eigenen Code hinzufügen, um Daten zu verarbeiten, die von S3 abgerufen werden, bevor Sie sie an eine Anwendung zurückgeben. Insbesondere können Sie eine - AWS Lambda Funktion konfigurieren und an einen S3 Object Lambda Access Point anfügen. Wenn eine Anwendung S3-GET-Standardanforderungen über den S3 Object Lambda Access Point sendet, wird die angegebene Lambda-Funktion aufgerufen, um alle Daten zu verarbeiten, die von einem S3-Bucket über den unterstützenden S3-Zugriffspunkt abgerufen werden. Dann gibt der S3 Object Lambda Access Point das transformierte Ergebnis zurück an die Anwendung aus. Sie können Ihre eigenen benutzerdefinierten Lambda-Funktionen erstellen und ausführen, indem Sie die S3 Object Lambda-Datentransformation an Ihren spezifischen Anwendungsfall anpassen, ohne dass Änderungen an Ihren Anwendungen erforderlich sind.



Ziel

In diesem Tutorial erfahren Sie, wie Sie benutzerdefinierten Code zu standardmäßigen S3-GET-Anforderungen hinzufügen, um das angeforderte Objekt, das von S3 abgerufen wurde, so zu ändern, dass das Objekt den Anforderungen des anfordernden Clients oder der anfordernden Anwendung entspricht. Insbesondere erfahren Sie, wie Sie den gesamten Text im ursprünglichen Objekt, das in S3 gespeichert ist, über S3 Object Lambda in Großbuchstaben umwandeln.

Themen

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, benötigen Sie ein AWS-Konto , bei dem Sie sich als AWS Identity and Access Management (IAM)-Benutzer mit den richtigen Berechtigungen anmelden können. Sie müssen außerdem Python Version 3.8 oder höher installieren.

Teilschritte

- Erstellen Sie einen IAM-Benutzer mit Berechtigungen in Ihrem AWS-Konto (Konsole)
- Installieren Sie Python 3.8 oder höher auf Ihrem lokalen Computer

Erstellen Sie einen IAM-Benutzer mit Berechtigungen in Ihrem AWS-Konto (Konsole)

Sie können für das Tutorial die Anmeldeinformationen eines IAM-Benutzers nutzen. Um dieses Tutorial abzuschließen, muss Ihr IAM-Benutzer die folgenden IAM-Richtlinien anfügen, um auf relevante AWS Ressourcen zuzugreifen und bestimmte Aktionen auszuführen. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen von IAM-Benutzern (Konsole) im IAM-Benutzerhandbuch.

Ihr IAM-Benutzer benötigt folgende Richtlinien:

- AmazonS3FullAccess – Gewährt Berechtigungen für alle Amazon S3-Aktionen, einschließlich Berechtigungen zum Erstellen und Verwenden eines Object Lambda Access Point.

- AWSLambda_FullAccess – Gewährt Berechtigungen für alle Lambda-Aktionen.
- IAMFullAccess – Gewährt Berechtigungen für alle IAM-Aktionen.
- IAMAccessAnalyzerReadOnlyAccess – Gewährt Berechtigungen zum Lesen aller von IAM Access Analyzer bereitgestellten Zugriffsinformationen.
- CloudWatchLogsFullAccess – Gewährt vollen Zugriff auf CloudWatch Protokolle.

> **Note**
>
> Der Einfachheit halber wird in diesem Tutorial einen IAM-Benutzer erstellt und verwendet. Denken Sie nach Abschluss dieses Tutorials an Löschen des IAM-Benutzers. Für den Einsatz in der Produktion empfehlen wir, dass Sie die bewährten Sicherheitsmethoden in IAM im IAM-Benutzerhandbuch befolgen. Eine bewährte Methode ist, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden, um mit temporären Anmeldeinformationen auf AWS zuzugreifen. Eine weitere bewährte Methode besteht darin, dass Workloads temporäre Anmeldeinformationen mit IAM-Rollen verwenden müssen, um auf AWS zuzugreifen. Weitere Informationen zur Verwendung von AWS IAM Identity Center zum Erstellen von Benutzern mit temporären Anmeldeinformationen finden Sie unter Erste Schritte im AWS IAM Identity Center -Benutzerhandbuch.
> In diesem Tutorial werden auch AWS -verwaltete Richtlinien mit vollem Zugriff verwendet. Für die Verwendung in der Produktion empfehlen wir, dass Sie stattdessen nur die für Ihren Anwendungsfall erforderlichen Mindestberechtigungen gemäß Bewährte Methoden in Bezug auf die Sicherheit erteilen.

Installieren Sie Python 3.8 oder höher auf Ihrem lokalen Computer

Verwenden Sie das folgende Verfahren, um Python 3.8 oder höher auf Ihrem lokalen Computer zu installieren. Weitere Installationsanweisungen finden Sie auf der Seite Python herunterladen im Python-Anfängerhandbuch.

1. Öffnen Sie das lokale Terminal oder die Shell, und führen Sie den folgenden Befehl aus, um festzustellen, ob Python bereits installiert ist, und wenn ja, welche Version installiert ist.

   ```
   python --version
   ```

2. Wenn Sie nicht über Python 3.8 oder höher verfügen, laden Sie die Datei offizieller Installer von Python 3.8 oder höher herunter, die für Ihren lokalen Computer geeignet ist.

3. Führen Sie das Installationsprogramm aus, indem Sie auf die heruntergeladene Datei doppelklicken und die Schritte ausführen, um die Installation abzuschließen.

   Wählen Sie für Windows-Nutzer Hinzufügen von Python 3.X zu PATH im Installationsassistenten, bevor Sie Jetzt installieren auswählen.
4. Starten Sie das Terminal neu, indem Sie es schließen und erneut öffnen.
5. Führen Sie den folgenden Befehl aus, um zu überprüfen, ob Python 3.8 oder höher ordnungsgemäß installiert ist.

   Führen Sie für macOS-Benutzer diesen Befehl aus:

   ```
   python3 --version
   ```

   Führen Sie für Windows-Benutzer diesen Befehl aus:

   ```
   python --version
   ```
6. Führen Sie den folgenden Befehl aus, um zu überprüfen, ob der pip3-Paket-Manager installiert ist. Wenn Sie in der Befehlsantwort eine Pip-Versionsnummer und Python 3.8 oder höher sehen, bedeutet dies, dass der pip3-Paketmanager erfolgreich installiert wurde.

   ```
   pip --version
   ```

Schritt 1: Einen S3-Bucket erstellen

Erstellen Sie einen Bucket zum Speichern der ursprünglichen Daten, die Sie transformieren möchten.

So erstellen Sie einen Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Die Seite Create bucket (Bucket erstellen) wird geöffnet.
4. Geben Sie für Bucket-Name einen Namen für Ihren Bucket ein (z. B. **`tutorial-bucket`**).

Weitere Informationen zur Benennung von Buckets in Amazon S3 finden Sie unter Regeln für die Benennung von Buckets.

5. Wählen Sie für Region die aus, AWS-Region in der sich der Bucket befinden soll.

   Weitere Informationen zur Bucket-Region finden Sie unter Bucket-Übersicht.

6. Belassen Sie die Einstellungen für den öffentlichen Zugriff für diesen Bucket blockieren bei den Standardeinstellungen (Alle öffentlichen Zugriffe blockieren ist aktiviert).

   Es wird empfohlen, alle Einstellungen für öffentlichen Zugriff blockieren aktiviert zu belassen, es sei denn, Sie müssen eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

7. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt.

   (Optional) Informationen zum Konfigurieren zusätzlicher Bucket-Einstellungen für Ihren spezifischen Anwendungsfall finden Sie unter Erstellen eines Buckets.

8. Wählen Sie Bucket erstellen aus.

Schritt 2: Hochladen einer Datei zu einem S3-Bucket

Laden Sie die Textdatei zu einem S3-Bucket hoch. Diese Textdatei enthält die Originaldaten, die Sie später in diesem Tutorial in Großbuchstaben umwandeln.

Sie können beispielsweise eine `tutorial.txt`-Datei, die den folgenden Text enthält, hochladen:

```
Amazon S3 Object Lambda Tutorial:
You can add your own code to process data retrieved from S3 before
returning it to an application.
```

Hochladen einer Datei in einen Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben, in den Sie die Datei hochladen möchten.

4. Wählen Sie im Tab Objects (Objekte) für Ihren Bucket die Option Upload (Hochladen) aus.
5. Wählen Sie auf der Seite Upload unter Dateien und Ordner die Option Dateien hinzufügen aus.
6. Wählen Sie eine hochzuladende Datei und dann Open (Öffnen) aus. Sie können z. B. das oben erwähnte `tutorial.txt`-Dateibeispiel hochladen.
7. Klicken Sie auf Hochladen.

Schritt 3: Erstellen eines S3-Zugriffspunkts

Um einen S3 Object Lambda Access Point für den Zugriff und die Umwandlung der ursprünglichen Daten zu verwenden, müssen Sie einen S3-Zugriffspunkt erstellen und dem S3-Bucket zuordnen, den Sie in Schritt 1 erstellt haben. Der Zugriffspunkt muss sich in derselben befinden AWS-Region wie die Objekte, die Sie transformieren möchten.

Später in diesem Tutorial verwenden Sie diesen Zugriffspunkt als unterstützenden Zugriffspunkt für Ihren Object Lambda Access Point.

So erstellen Sie einen Zugangspunkt

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffspunkte aus.
3. Wählen Sie auf der Seite Zugriffspunkte die Option Zugriffspunkt erstellen aus.
4. Geben Sie im Feld Name des Zugriffspunkts den gewünschten Namen für den Zugriffspunkt ein (z. B. **`tutorial-access-point`**).

   Weitere Hinweise zur Benennung von Zugangspunkten finden Sie unter Regeln zur Benennung von Amazon S3-Zugriffspunkten.
5. Wählen Sie im Feld Bucket-Name den Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben. S3 hängt den Zugriffspunkt an diesen Bucket an.

   (Optional) Sie können S3 durchsuchen auswählen, um die Buckets in Ihrem Konto zu browsen und zu durchsuchen. Wenn Sie Browse S3 (S3 durchsuchen) wählen, wählen Sie den gewünschten Bucket aus und dann Choose path (Pfad wählen), um das Feld Bucket name (Bucket-Name) mit dem Namen dieses Buckets zu füllen.
6. Wählen Sie für Netzwerkursprung Internetaus.

Weitere Informationen zu Netzwerkursprüngen für Zugangspunkte finden Sie unter Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind.

7. Alle Block-Public-Access-Einstellungen sind standardmäßig für Zugriffspunkte aktiviert. Sie sollten alle Einstellungen für das Blockieren aller öffentlichen Zugriffe aktiviert lassen.

   Weitere Informationen finden Sie unter Verwalten des öffentlichen Zugriffs auf Zugriffspunkte.

8. Behalten Sie für alle anderen Zugriffspunktseinstellungen die Standardeinstellungen bei.

   (Optional) Sie können die Zugriffspunktseinstellungen ändern, um Ihren Anwendungsfall zu unterstützen. Für dieses Tutorial empfehlen wir, dass die Standardeinstellungen beibehalten werden.

   (Optional) Wenn Sie den Zugriff auf Ihren Zugriffspunkt verwalten müssen, können Sie eine Zugriffspunkt-Richtlinie angeben. Weitere Informationen finden Sie unter Beispiele von - Zugriffspunktrichtlinien.

9. Wählen Sie Create access point (Zugriffspunkt erstellen) aus.

Schritt 4: Erstellen einer Lambda-Funktion

Erstellen Sie zum Transformieren der ursprünglichen Daten eine Lambda-Funktion zur Verwendung mit dem S3 Object Lambda Access Point.

Teilschritte

- Schreiben von Lambda-Funktionscode und Erstellen eines Bereitstellungspakets mit einer virtuellen Umgebung
- Erstellen einer Lambda-Funktion mit einer Ausführungsrolle (Konsole)
- Stellen Sie Ihren Lambda-Funktion-Code mit ZIP-Dateiarchiven bereit und konfigurieren Sie die Lambda-Funktion (Konsole)

Schreiben von Lambda-Funktionscode und Erstellen eines Bereitstellungspakets mit einer virtuellen Umgebung

1. Erstellen Sie auf dem lokalen Computer einen Ordner mit dem Ordnernamen `object-lambda` für die virtuelle Umgebung, die Sie zu einem späteren Zeitpunkt in diesem Tutorial verwenden können.

2. Erstellen Sie im `object-lambda`-Ordner eine Datei mit einer Lambda-Funktion, die den gesamten Text im ursprünglichen Objekt in Großbuchstaben ändert. Sie können z. B. die folgende Funktion verwenden, die in Python geschrieben wurde. Speichern Sie diese Funktion in einer Datei mit dem Name`transform.py`.

```
import boto3
import requests
from botocore.config import Config

# This function capitalizes all text in the original object
def lambda_handler(event, context):
    object_context = event["getObjectContext"]
    # Get the presigned URL to fetch the requested original object
    # from S3
    s3_url = object_context["inputS3Url"]
    # Extract the route and request token from the input context
    request_route = object_context["outputRoute"]
    request_token = object_context["outputToken"]

    # Get the original S3 object using the presigned URL
    response = requests.get(s3_url)
    original_object = response.content.decode("utf-8")

    # Transform all text in the original object to uppercase
    # You can replace it with your custom code based on your use case
    transformed_object = original_object.upper()

    # Write object back to S3 Object Lambda
    s3 = boto3.client('s3', config=Config(signature_version='s3v4'))
    # The WriteGetObjectResponse API sends the transformed data
    # back to S3 Object Lambda and then to the user
    s3.write_get_object_response(
        Body=transformed_object,
        RequestRoute=request_route,
        RequestToken=request_token)

    # Exit the Lambda function: return the status code
    return {'status_code': 200}
```

> **Note**
>
> Die vorangehende Lambda-Beispielfunktion lädt das gesamte angeforderte Objekt in den Speicher, bevor es transformiert und an den Client zurückgegeben wird. Alternativ können Sie das Objekt aus S3 streamen, um das Laden des gesamten Objekts in den Speicher zu vermeiden. Diese Methode ist nützlich, wenn Sie mit großen Objekten arbeiten. Weitere Informationen zum Streamen von Antworten mit Object Lambda Access Points finden Sie in den Streaming-Beispielen in Arbeiten mit `GetObject`-Anforderungen in Lambda.

Wenn Sie eine Lambda-Funktion für die Verwendung mit einem S3 Object Lambda Access Point schreiben, basiert die Funktion auf dem Eingabe-Ereigniskontext, den S3 Object Lambda für die Lambda-Funktion bereitstellt. Der Ereigniskontext stellt Informationen über die Anforderung bereit, die in dem von S3 Object Lambda an Lambda übergebenen Ereignis gestellt wird. Sie enthält die Parameter, die Sie verwenden, um die Lambda-Funktion zu erstellen.

Die Felder, die zum Erstellen der vorhergehenden Lambda-Funktion verwendet werden, lauten wie folgt:

Das Feld von `getObjectContext` bezeichnet die Eingabe- und Ausgabedetails für Verbindungen zu Amazon S3 und S3 Object Lambda. Es enthält die folgenden Felder:

- `inputS3Url` – Eine vorsignierte URL, mit der die Lambda-Funktion das ursprüngliche Objekt vom unterstützenden Zugriffspunkt herunterladen kann. Wenn Sie eine vorsignierte URL verwenden, benötigt die Lambda-Funktion keine Amazon S3-Leseberechtigungen, um das ursprüngliche Objekt abzurufen, und kann nur auf das Objekt zugreifen, das von jedem Aufruf verarbeitet wird.
- `outputRoute` – Ein Routing-Token, das der Lambda-URL des S3-Objekts hinzugefügt wird, wenn die Lambda-Funktion `WriteGetObjectResponse` aufruft, um das transformierte Objekt zurückzusenden.
- `outputToken` – Ein Token, das von S3 Object Lambda verwendet wird, um den `WriteGetObjectResponse`-Aufruf mit dem ursprünglichen Anrufer abzugleichen, wenn das transformierte Objekt zurücksendet wird.

Weitere Informationen zu allen Feldern im Ereigniskontext finden Sie unter Format und Verwendung des Ereigniskontexts und Schreiben von Lambda-Funktionen für S3 Object Lambda Access Points.

3. Geben Sie in Ihrem lokalen Terminal den folgenden Befehl ein, um das `virtualenv`-Paket zu installieren:

```
python -m pip install virtualenv
```

4. Öffnen Sie in Ihrem lokalen Terminal den `object-lambda`-Ordner, den Sie zuvor erstellt haben und geben Sie dann den folgenden Befehl ein, um eine virtuelle Umgebung namens `venv` zu erstellen und zu initialisieren.

```
python -m virtualenv venv
```

5. Geben Sie zum Aktivieren der virtuellen Umgebung den folgenden Befehl ein, um die `activate`-Datei aus dem Ordner der Umgebung auszuführen:

   Führen Sie für macOS-Benutzer diesen Befehl aus:

```
source venv/bin/activate
```

   Führen Sie für Windows-Benutzer diesen Befehl aus:

```
.\venv\Scripts\activate
```

   Jetzt ändert sich die Eingabeaufforderung und zeigt (venv) an, was darauf hinweist, dass die virtuelle Umgebung aktiv ist.

6. Um die erforderlichen Bibliotheken zu installieren, führen Sie die folgenden Befehle Zeile für Zeile in der virtuellen `venv`-Umgebung aus.

   Mit diesen Befehlen werden aktualisierte Versionen der Abhängigkeiten Ihrer `lambda_handler`-Lambda -Funktion installiert. Diese Abhängigkeiten sind die AWS -SDK for Python (Boto3)- und Requests-Module.

```
pip3 install boto3
```

```
pip3 install requests
```

7. Sie können die virtuelle Umgebung deaktivieren, indem Sie den folgenden Befehl ausführen:

```
deactivate
```

8. Um ein Bereitstellungspaket mit den installierten Bibliotheken als `.zip`-Datei mit dem Namen `lambda.zip` an der Wurzel des `object-lambda`-Verzeichnisses zu erstellen, führen Sie die folgenden Befehle Zeile für Zeile in Ihrem lokalen Terminal aus.

> **Tip**
>
> Die folgenden Befehle müssen möglicherweise angepasst werden, um in Ihrer bestimmten Umgebung zu funktionieren. Eine Bibliothek kann beispielsweise in `site-packages` oder `dist-packages` erscheinen, und der erste Ordner könnte `lib` oder `lib64` sein. Außerdem kann der Ordner `python` mit einer anderen Python-Version benannt werden. Sie können den Befehl `pip show` verwenden, um ein spezielles Paket zu finden.

Führen Sie für macOS-Benutzer diese Befehle aus:

```
cd venv/lib/python3.8/site-packages
```

```
zip -r ../../../../lambda.zip .
```

Führen Sie für Windows-Benutzer diese Befehle aus:

```
cd .\venv\Lib\site-packages\
```

```
powershell Compress-Archive * ../../../lambda.zip
```

Der letzte Befehl speichert das Bereitstellungspaket im Stamm des `object-lambda`-Verzeichnisses.

9. Fügen Sie dem Stamm Ihres Bereitstellungspakets die Funktionscode-Datei `transform.py` hinzu.

Führen Sie für macOS-Benutzer diese Befehle aus:

```
cd ../../../../
```

```
zip -g lambda.zip transform.py
```

Führen Sie für Windows-Benutzer diese Befehle aus:

```
cd ..\..\..\
```

```
powershell Compress-Archive -update transform.py lambda.zip
```

Nach diesem Schritt sollte Ihre Verzeichnisstruktur wie folgt aussehen:

```
lambda.zip$
  # transform.py
  # __pycache__
  | boto3/
  # certifi/
  # pip/
  # requests/
  ...
```

Erstellen einer Lambda-Funktion mit einer Ausführungsrolle (Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie im linken Navigationsbereich die Option Functions (Funktionen) aus.
3. Wählen Sie Create function (Funktion erstellen).
4. Wählen Sie Author from scratch aus.
5. Führen Sie unter Basic information (Grundlegende Informationen) die folgenden Schritte aus:
   a. Geben Sie für Function name (Funktionsname) **`tutorial-object-lambda-function`** ein.
   b. Wählen Sie für Laufzeit Python 3.8 oder eine höhere Version aus.

6. Erweitern Sie den Abschnitt Ändern der standardmäßigen Ausführungsrolle. Wählen Sie in Execution role (Ausführungsrolle) die Option Create a new role with basic Lambda permissions (Neue Rolle mit Lambda-Basisberechtigungen erstellen) aus.

   In Schritt 5 weiter unten in diesem Tutorial fügen Sie die AmazonS3ObjectLambdaExecutionRolePolicy an die Ausführungsrolle dieser Lambda-Funktion an.

7. Lassen Sie die restlichen Einstellungen bei den Standardwerten.
8. Wählen Sie Funktion erstellen.

Stellen Sie Ihren Lambda-Funktion-Code mit ZIP-Dateiarchiven bereit und konfigurieren Sie die Lambda-Funktion (Konsole)

1. Wählen Sie in der AWS Lambda -Konsole unter https://console.aws.amazon.com/lambda/ im linken Navigationsbereich Funktionen aus.
2. Wählen Sie die Lambda-Funktion aus, die Sie zuvor erstellt haben (z. B **`tutorial-object-lambda-function`**).
3. Wählen Sie auf der Seite mit den Details der Lambda-Funktion den Code-Tab. Wählen Sie im Abschnitt Quellcode die Option Upload von und dann ZIP-Datei aus.
4. Wählen Sie Upload (Hochladen) aus, um Ihre lokale `.zip`-Datei auszuwählen.
5. Wählen Sie die zuvor erstellte `lambda.zip`-Datei und klicken Sie dann auf Öffnen.
6. Wählen Sie Save (Speichern) aus.
7. Wählen Sie im Abschnitt Laufzeiteinstellungen die OptionBearbeiten aus.
8. Bestätigen Sie auf der Seite Bearbeiten von Laufzeiteinstellungen, dass Laufzeit auf Python 3.8 oder eine höhere Version gesetzt wird.
9. Um der Lambda-Laufzeit mitzuteilen, welche Handler-Methode in Ihrem Lambda-Funktionscode aufgerufen werden soll, geben Sie **`transform.lambda_handler`** für Handler ein.

   Wenn Sie eine Funktion in Python konfigurieren, besteht der Wert der Handler-Einstellung aus dem Dateinamen und dem Namen des Handler-Moduls, getrennt durch einen Punkt. Beispielsweise ruft `transform.lambda_handler` die `lambda_handler`-Methode auf, die in der `transform.py`-Datei definiert ist.

10. Wählen Sie Speichern.

11. (Optional) Wählen Sie auf der Detailseite der Lambda-Funktion den Konfiguration-Tab. Wählen Sie im linken Navigationsbereich die OptionAllgemeine Konfiguration und anschließend Bearbeiten aus. Geben Sie im Timeout-Feld **1** m **0** s ein. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt und wählen Sie Speichern aus.

    Timeout ist die Zeitspanne, die Lambda einer Funktion für einen Aufruf zulässt, bevor diese gestoppt wird. Der Standardwert ist 3 Sekunden. Die maximale Dauer für eine Lambda-Funktion, die von S3 Object Lambda verwendet wird, beträgt 60 Sekunden. Die Preise basieren auf dem konfigurierten Arbeitsspeicher und der Zeit, für die der Code ausgeführt wird.

Schritt 5: Konfigurieren einer IAM-Richtlinie für die Ausführungsrolle Ihrer Lambda-Funktion

Damit Ihre Lambda-Funktion benutzerdefinierte Daten- und Antwort-Header für den `GetObject`-Anrufer aktiviert, muss die Ausführungsrolle Ihrer Lambda-Funktion IAM-Berechtigungen besitzen, um die `WriteGetObjectResponse`-API anzurufen.

So fügen Sie eine IAM-Richtlinie an Ihre Lambda-Funktionsrolle an

1. Wählen Sie in der AWS Lambda -Konsole unter https://console.aws.amazon.com/lambda/ im linken Navigationsbereich Funktionen aus.
2. Wählen Sie die Funktion aus, die Sie in Schritt 4 erstellt haben (z. B. **`tutorial-object-lambda-function`**).
3. Wählen Sie auf der Detailseite Ihrer Lambda-Funktion den Tag Konfiguration und danach Berechtigungen im linken Navigationsbereich.
4. Wählen Sie unter Ausführungsrolle im Bereich den Link des Rollennamens aus. Die IAM-Konsole wird geöffnet.
5. Wählen Sie auf der Seite Summary (Übersicht) der IAM-Konsole für die Ausführungsrolle Ihrer Lambda-Funktion die Registerkarte Permissions (Berechtigungen) aus. Wählen Sie dann im Menü Add Permissions (Berechtigungen hinzufügen) die Option Attach policies (Richtlinien anfügen) aus.
6. Geben Sie auf der Seite Attach permissions (Berechtigungen anfügen) **`AmazonS3ObjectLambdaExecutionRolePolicy`** in das Suchfeld ein, um die Liste der Richtlinien zu filtern. Aktivieren Sie das Kontrollkästchen neben dem Namen der AmazonS3ObjectLambdaExecutionRolePolicy-Richtlinie.

7. Wählen Sie Richtlinien anfügen.

Schritt 6: Erstellen eines S3 Object Lambda Access Point

Ein S3 Object Lambda Access Point bietet die Flexibilität, eine Lambda-Funktion direkt aus einer S3-GET-Anforderung aufzurufen, sodass die Funktion Daten verarbeiten kann, die von einem S3-Zugriffspunkt abgerufen wurden. Beim Erstellen und Konfigurieren eines S3 Object Lambda Access Point müssen Sie die Lambda-Funktion angeben, um den Ereigniskontext im JSON-Format als benutzerdefinierte Parameter für Lambda zu verwenden.

So erstellen Sie einen S3 Object Lambda Access Point

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie auf der Seite Object Lambda access points (Objekt-Lambda-Zugriffspunkte) die Option Create Object Lambda access point (Objekt-Lambda-Zugriffspunkt erstellen).
4. Geben Sie unter Name des Objekt-Lambda-Access Point den Namen ein, den Sie für den Object Lambda Access Point verwenden möchten (z. B. **`tutorial-object-lambda-accesspoint`**).
5. Geben Sie für Unterstützenden Zugangspunkt den Standardzugangspunkt ein, den Sie in Schritt 3 erstellt haben (z. B. **`tutorial-access-point`**), oder navigieren Sie zu diesem Zugangspunkt und wählen dann Unterstützenden Zugangspunkt auswählen aus.
6. Um für S3-APIs Objekte aus dem S3-Bucket abzurufen, die die Lambda-Funktion verarbeiten soll, wählen Sie ausGetObject.
7. Für die Funktion Lambda aufrufen können Sie eine der beiden folgenden Optionen für dieses Tutorial auswählen.
   - Wählen Sie Wählen Sie aus Funktionen in Ihrem Kontound wählen Sie die Lambda-Funktion aus, die Sie in Schritt 4 (z. B. **`tutorial-object-lambda-function`**) aus der Lambda-Funktion-Dropdown-Liste erstellt haben.
   - Wählen Sie ARN eingeben und geben Sie dann den Amazon-Ressourcennamen (ARN) der Lambda-Funktion ein, die Sie in Schritt 4 erstellt haben.
8. Wählen Sie für Versioning der Lambda-Funktion, $LATEST (die neueste Version der Lambda-Funktion, die Sie in Schritt 4 erstellt haben).
9. (Optional) Wenn Sie Ihre Lambda-Funktion benötigen, um GET-Anforderungen mit Bereichs- und Teilenummern-Headern zu erkennen und zu bearbeiten, wählen Sie Lambda-Funktion

unterstützt Anforderungen mit Bereich und Lambda-Funktion unterstützt Anforderungen mit Teilenummern aus. Deaktivieren Sie andernfalls diese beiden Kontrollkästchen.

Weitere Informationen zur Verwendung von Bereichs- oder Teilenummern mit S3 Object Lambda finden Sie unter Arbeiten mit Range- und partNumber-Headern.

10. (Optional) Fügen Sie unter Nutzlast – optional JSON-Text hinzu, um Ihrer Lambda-Funktion zusätzliche Informationen bereitzustellen.

    Eine Nutzlast ist ein optionaler JSON-Text, den Sie Ihrer Lambda-Funktion als Eingabe für alle Aufrufe bereitstellen können, die von einem spezifischen S3 Object Lambda Access Point stammen. Um das Verhalten für mehrere Object Lambda Access Points anzupassen, die dieselbe Lambda-Funktion aufrufen, können Sie Nutzlasten mit unterschiedlichen Parametern konfigurieren und so die Flexibilität Ihrer Lambda-Funktion erweitern.

    Weitere Informationen zur Nutzlast finden Sie unter Format und Verwendung des Ereigniskontexts.

11. (Optional) Wählen Sie für Anforderungsmetriken – optional die Option Deaktivieren oder Aktivieren aus, um Ihrem Object Lambda Access Point Amazon-S3-Überwachung hinzuzufügen. Anforderungsmetriken werden zum Amazon- CloudWatch Standardtarif abgerechnet. Weitere Informationen finden Sie unter CloudWatch Preise.

12. Behalten Sie unter Objekt-Lambda-Zugriffspunkt-Richtlinie - optional die Standardeinstellung bei.

    (Optional) Sie können eine Ressourcenrichtlinie festlegen. Diese Ressourcenrichtlinie erteilt der `GetObject`-API die Berechtigung zur Verwendung des angegebenen Object Lambda Access Point.

13. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt und klicken Sie auf Erstellen eines Objekt-Lambda-Zugriffspunkts.

Schritt 7: Anzeigen der transformierten Daten

Jetzt ist S3 Object Lambda bereit, Ihre Daten für Ihren Anwendungsfall zu transformieren. In diesem Tutorial wandelt S3 Object Lambda den gesamten Text in Ihrem Objekt in Großbuchstaben um.

Teilschritte

- Zeigen Sie die transformierten Daten in Ihrem S3 Object Lambda Access Point an
- Ausführen eines Python-Skripts zum Drucken der ursprünglichen und transformierten Daten

Zeigen Sie die transformierten Daten in Ihrem S3 Object Lambda Access Point an

Wenn Sie eine Datei über Ihren S3 Object Lambda Access Point abrufen möchten, führen Sie einen `GetObject`-API-Aufruf für S3 Object Lambda aus. S3 Object Lambda ruft die Lambda-Funktion auf, um Ihre Daten zu transformieren und gibt dann die transformierten Daten als Antwort auf die Standard-S3-`GetObject` API-Aufrufe.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie auf der Seite Objekt-Lambda-Zugriffspunkte den S3 Object Lambda Access Point aus, den Sie in Schritt 6 erstellt haben (z. B. **`tutorial-object-lambda-accesspoint`**).
4. Wählen Sie auf der Registerkarte Objekte Ihres S3 Object Lambda Access Point die Datei mit dem gleichen Namen (z. B. `tutorial.txt`) wie diejenige aus, die Sie in Schritt 2 in den S3-Bucket hochgeladen haben.

   Diese Datei sollte alle transformierten Daten enthalten.
5. Um die transformierten Daten anzuzeigen, wählen Sie Öffnen oder Herunterladen aus.

Ausführen eines Python-Skripts zum Drucken der ursprünglichen und transformierten Daten

Sie können S3 Object Lambda mit Ihren vorhandenen Anwendungen verwenden. Aktualisieren Sie dazu Ihre Anwendungskonfiguration, um den ARN des neuen S3 Object Lambda Access Point zu verwenden, den Sie in Schritt 6 erstellt haben, um Daten aus S3 abzurufen.

Das folgende Python-Beispielskript druckt sowohl die ursprünglichen Daten aus dem S3-Bucket als auch die transformierten Daten aus dem S3 Object Lambda Access Point.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie auf der Seite Objekt-Lambda-Zugriffspunkte die Optionsschaltfläche links neben dem S3 Object Lambda Access Point aus, den Sie in Schritt 6 erstellt haben (z. B. **`tutorial-object-lambda-accesspoint`**).
4. Klicken Sie auf ARN kopieren.

5. Speichern Sie den ARN zur späteren Verwendung.
6. Schreiben Sie ein Python-Skript auf Ihrem lokalen Computer, um sowohl die ursprünglichen Daten aus Ihrem S3-Bucket (z. B. `tutorial.txt`) als auch die transformierten Daten aus Ihrem S3 Object Lambda Access Point (z. B. `tutorial.txt`) zu drucken. Sie können folgendes Beispiel-Skript nutzen.

```
import boto3
from botocore.config import Config

s3 = boto3.client('s3', config=Config(signature_version='s3v4'))

def getObject(bucket, key):
    objectBody = s3.get_object(Bucket = bucket, Key = key)
    print(objectBody["Body"].read().decode("utf-8"))
    print("\n")

print('Original object from the S3 bucket:')
# Replace the two input parameters of getObject() below with
# the S3 bucket name that you created in Step 1 and
# the name of the file that you uploaded to the S3 bucket in Step 2
getObject("tutorial-bucket",
          "tutorial.txt")

print('Object transformed by S3 Object Lambda:')
# Replace the two input parameters of getObject() below with
# the ARN of your S3 Object Lambda Access Point that you saved earlier and
# the name of the file with the transformed data (which in this case is
# the same as the name of the file that you uploaded to the S3 bucket
# in Step 2)
getObject("arn:aws:s3-object-lambda:us-west-2:111122223333:accesspoint/tutorial-
object-lambda-accesspoint",
          "tutorial.txt")
```

7. Speichern Sie Ihr Python-Skript unter einem benutzerdefinierten Namen (z. B. `tutorial_print.py`) im Ordner (z. B. `object-lambda`), den Sie in Schritt 4 auf Ihrem lokalen Computer erstellt haben.
8. Führen Sie in Ihrem lokalen Terminal den folgenden Befehl vom Stamm des Verzeichnisses aus (z. B. `object-lambda`), das Sie inSchritt 4 erstellt haben.

```
python3 tutorial_print.py
```

Sie sollten sowohl die ursprünglichen Daten als auch die transformierten Daten (alle Texte in Großbuchstaben) über das Terminal sehen. z. B. sollten Sie etwas wie den folgenden Text sehen.

```
Original object from the S3 bucket:
Amazon S3 Object Lambda Tutorial:
You can add your own code to process data retrieved from S3 before
returning it to an application.

Object transformed by S3 Object Lambda:
AMAZON S3 OBJECT LAMBDA TUTORIAL:
YOU CAN ADD YOUR OWN CODE TO PROCESS DATA RETRIEVED FROM S3 BEFORE
RETURNING IT TO AN APPLICATION.
```

Schritt 8: Bereinigen

Wenn Sie Ihre Daten über S3 Object Lambda nur als Tutorial transformiert haben, löschen Sie die AWS -Ressourcen, die Sie zugewiesen haben, damit keine Gebühren mehr anfallen.

Teilschritte

- Löschen des Object Lambda Access Point
- Löschen des S3-Zugriffspunkts
- Löschen der Ausführungsrolle für Ihre Lambda-Funktion
- Löschen Sie die Lambda-Funktion
- Löschen der CloudWatch Protokollgruppe
- Löschen Sie die Originaldatei im S3-Quell-Bucket
- Löschen des S3-Quell-Bucket
- Löschen des IAM-Benutzers

Löschen des Object Lambda Access Point

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.

3. Wählen Sie auf der Seite Objekt-Lambda-Zugriffspunkte die Optionsschaltfläche links neben dem S3 Object Lambda Access Point aus, den Sie in Schritt 6 erstellt haben (z. B. **`tutorial-object-lambda-accesspoint`**).
4. Wählen Sie Delete (Löschen).
5. Bestätigen Sie, dass Sie Ihren Object Lambda Access Point löschen möchten, indem Sie den Namen in das angezeigte Textfeld eingeben und dann Löschen wählen.

Löschen des S3-Zugriffspunkts

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffspunkte aus.
3. Navigieren Sie zu dem Zugriffspunkt, den Sie in Schritt 3 (z. B. **`tutorial-access-point`**) erstellt haben, und wählen Sie die Schaltfläche neben dem Namen des Zugriffspunkts aus.
4. Wählen Sie Löschen aus.
5. Bestätigen Sie, dass Sie Ihren Zugriffspunkt löschen möchten, indem Sie den Namen in das angezeigte Textfeld eingeben und dann Delete (Löschen) wählen.

Löschen der Ausführungsrolle für Ihre Lambda-Funktion

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie im linken Navigationsbereich die Option Funktionen aus.
3. Wählen Sie die Funktion aus, die Sie in Schritt 4 erstellt haben (z. B. **`tutorial-object-lambda-function`**).
4. Wählen Sie auf der Detailseite Ihrer Lambda-Funktion den Tag Konfiguration und danach Berechtigungen im linken Navigationsbereich.
5. Wählen Sie unter Ausführungsrolle im Bereich den Link des Rollennamens aus. Die IAM-Konsole wird geöffnet.
6. Wählen Sie auf der Seite Übersicht der IAM-Konsole Ihrer Ausführungsrolle der Lambda-Funktion Rolle löschen aus.
7. Klicken Sie im Dialogfeld auf Rolle löschen und anschließend auf Ja, löschen.

Löschen Sie die Lambda-Funktion

1. Wählen Sie in der AWS Lambda -Konsole unter https://console.aws.amazon.com/lambda/ im linken Navigationsbereich Funktionen aus.
2. Aktivieren Sie das Kontrollkästchen links neben dem Namen der Funktion, die Sie im Abschnitt Schritt 4 (z. B. **`tutorial-object-lambda-function`**) erstellt haben.
3. Wählen Sie Actions (Aktionen) und anschließend Delete (Löschen).
4. Wählen Sie im Bestätigungsdialogfeld Delete (Löschen) die Option Delete (Löschen) aus.

Löschen der CloudWatch Protokollgruppe

1. Öffnen Sie die - CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Log groups (Protokollgruppen) aus.
3. Suchen Sie die Protokollgruppe, deren Name mit der Lambda-Funktion endet, die Sie in Schritt 4 (z. B. **`tutorial-object-lambda-function`**) erstellt haben.
4. Aktivieren Sie das Kontrollkästchen links neben dem Namen der Protokollgruppe.
5. Wählen Sie Actions (Aktionen) und dann Delete log group(s) (Protokollgruppe(n) löschen) aus.
6. Wählen Sie im Dialogfeld Delete log group(s) (Protokollgruppe(n) löschen) die Option Delete (Löschen) aus.

Löschen Sie die Originaldatei im S3-Quell-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Bucket-Name den Namen des Buckets aus, in den Sie die Originaldatei in Schritt 2 (z. B. **`tutorial-bucket`**) hochgeladen haben.
4. Markieren Sie das Kontrollkästchen links neben dem Namen des Objekts, das Sie löschen möchten (z. B. `tutorial.txt`).
5. Wählen Sie Löschen aus.
6. Bestätigen Sie auf der Seite Löschen von Objekten im Abschnitt Objekte endgültig löschen?, dass Sie dieses Objekt löschen möchten, indem Sie **`permanently delete`** im Textfeld eingeben.

7. Wählen Sie Delete objects (Objekte löschen).

Löschen des S3-Quell-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets die Optionsschaltfläche neben dem Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.
4. Wählen Sie Löschen aus.
5. Bestätigen Sie auf der Seite Delete bucket (Bucket löschen), dass Sie den Bucket löschen möchten. Geben Sie dazu den Bucket-Namen in das Textfeld ein und wählen Sie Delete bucket (Bucket löschen).

Löschen des IAM-Benutzers

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Benutzer, und aktivieren Sie dann das Kontrollkästchen neben dem Benutzer, den Sie löschen möchten.
3. Wählen Sie oben auf der Seite Delete (Löschen) aus.
4. Geben Sie im Dialogfeld ***`Benutzernamen`*** löschen? den Benutzernamen in das Texteingabefeld ein, um das Löschen des Benutzers zu bestätigen. Wählen Sie Delete (Löschen) aus.

Nächste Schritte

Nach Abschluss dieses Tutorials können Sie die Lambda-Funktion für Ihren Anwendungsfall anpassen, um die von Standard-S3-GET-Anforderungen zurückgegebenen Daten zu ändern.

Im Folgenden finden Sie eine Liste der häufigsten Anwendungsfälle für S3 Object Lambda:

- Maskierung sensibler Daten für Sicherheit und Compliance.

  Weitere Informationen finden Sie unter Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend.

- Filtern bestimmter Datenzeilen, um bestimmte Informationen bereitzustellen.
- Erweitern von Daten mit Informationen aus anderen Diensten oder Datenbanken.
- Konvertierung über Datenformate hinweg, z. B. das Konvertieren von XML in JSON zur Anwendungskompatibilität.
- Komprimieren oder Dekomprimieren von Dateien, während sie heruntergeladen werden.
- Größenänderung und Wasserzeichen für Bilder.

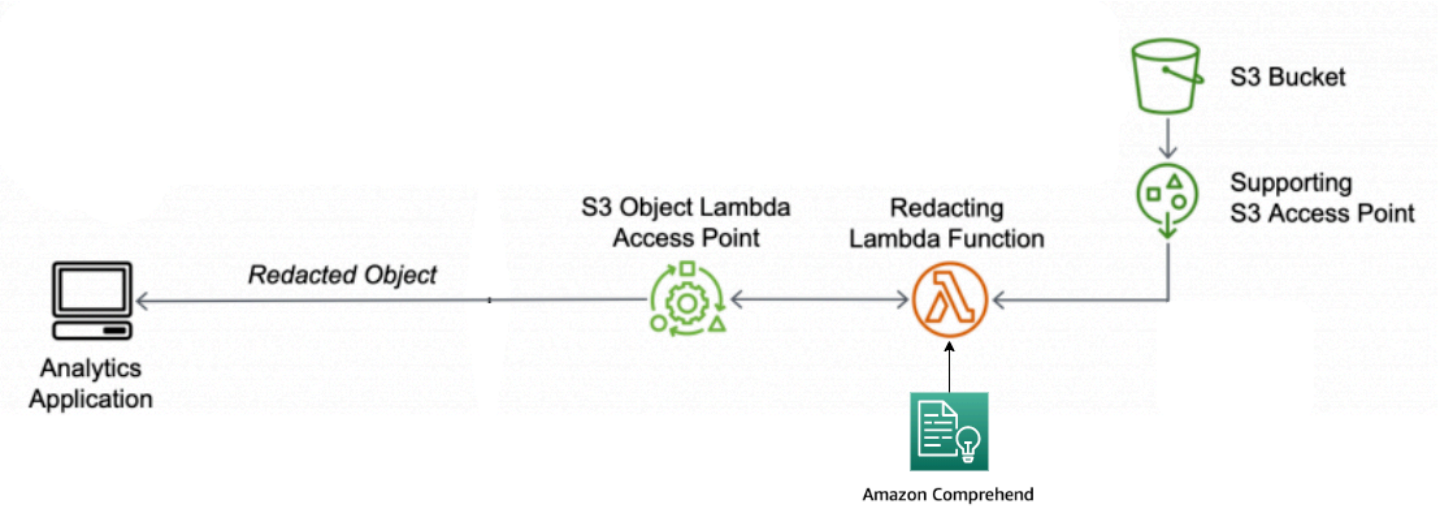
  Weitere Informationen finden Sie unter Tutorial: Verwenden von S3 Object Lambda, um Bilder beim Abrufen dynamisch mit Wasserzeichen zu versehen.
- Implementieren benutzerdefinierter Autorisierungsregeln für den Zugriff auf Daten.

Weitere Informationen zu S3 Object Lambda finden Sie unter Transformieren von Objekten mit S3 Object Lambda.

Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend

Wenn Sie Amazon S3 für gemeinsam genutzte Datensätze für mehrere Anwendungen und Benutzer verwenden, die darauf zugreifen können, ist es wichtig, privilegierte Informationen wie persönlich identifizierbare Informationen (PII) nur auf autorisierte Entitäten zu beschränken. Wenn beispielsweise eine Marketing-Anwendung einige Daten verwendet, die PII enthalten, müssen möglicherweise zuerst PII-Daten maskiert werden, um die Datenschutzanforderungen zu erfüllen. Wenn eine Analytikanwendung ein Bestandsdatensatz für Produktionsaufträge verwendet, muss sie möglicherweise zuerst die Kreditkarteninformationen des Kunden überarbeiten, um unbeabsichtigte Datenverluste zu verhindern.

Mit S3 Object Lambda und eine vorgefertigte AWS Lambda -Funktion, die von Amazon Comprehend unterstützt wird, können Sie PII-Daten schützen, die von S3 abgerufen wurden, bevor Sie sie an eine Anwendung zurückgeben. Insbesondere können Sie die vorgefertigte Lambda-Funktion als Redigierungsfunktion nutzen und sie an einen S3 Object Lambda Access Point anfügen. Wenn eine Anwendung (z. B. eine Analytik-Anwendung) S3-GET-Standardanforderungen sendet, rufen diese Anforderungen über den S3 Object Lambda Access Point die vordefinierte redigierende Lambda-Funktion auf, um PII-Daten, die von einem S3-Bucket über einen unterstützenden S3-Zugriffspunkt abgerufen wurden, zu erkennen und zu redigieren. Dann gibt der S3 Object Lambda Access Point das redigierte Ergebnis zurück an die Anwendung aus.



In diesem Prozess verwendet die vorgefertigte Lambda-Funktion Amazon Comprehend, eine natürliche Sprachverarbeitung (NLP), um Variationen in der Darstellung von PII zu erfassen, unabhängig davon, wie PII im Text vorhanden sind (z. B. numerisch oder als Kombination aus Wörtern und Zahlen). Amazon Comprehend kann sogar Kontext im Text verwenden, um zu verstehen, ob eine vierstellige Zahl eine PIN, die letzten vier Zahlen einer Sozialversicherungsnummer (SSN) oder ein Jahr ist. Amazon Comprehend verarbeitet jede Textdatei im UTF-8-Format und kann PII skaliert schützen, ohne die Genauigkeit zu beeinträchtigen. Weitere Informationen finden Sie unter Was ist Amazon Comprehend? im Entwicklerhandbuch für Amazon Comprehend.

Ziel

In diesem Tutorial erfahren Sie, wie Sie S3 Object Lambda mit der vordefinierten Lambda-Funktion `ComprehendPiiRedactionS3ObjectLambda` verwenden. Diese Funktion verwendet Amazon Comprehend, um PII-Entitäten zu erkennen. Diese Entitäten werden dann durch Sternchen ersetzt. Durch das Redigieren von PII verbergen Sie sensible Daten, die bei Sicherheit und Compliance helfen können.

Sie erfahren auch, wie Sie eine vorgefertigte AWS Lambda Funktion in der verwenden und konfigurierenAWS Serverless Application Repository, um mit S3 Object Lambda für eine einfache Bereitstellung zu arbeiten.

Themen

- Voraussetzungen: Erstellen Sie einen IAM-Benutzer mit Berechtigungen
- Schritt 1: Einen S3-Bucket erstellen
- Schritt 2: Hochladen einer Datei zu einem S3-Bucket

Voraussetzungen: Erstellen Sie einen IAM-Benutzer mit Berechtigungen

Bevor Sie mit diesem Tutorial beginnen, benötigen Sie ein - AWS Konto, bei dem Sie sich als - AWS Identity and Access Management Benutzer (IAM-Benutzer) mit den richtigen Berechtigungen anmelden können.

Sie können für das Tutorial die Anmeldeinformationen eines IAM-Benutzers nutzen. Um dieses Tutorial abzuschließen, muss Ihr IAM-Benutzer die folgenden IAM-Richtlinien anfügen, um auf relevante AWS Ressourcen zuzugreifen und bestimmte Aktionen auszuführen.

> **Note**
>
> Der Einfachheit halber wird in diesem Tutorial einen IAM-Benutzer erstellt und verwendet. Denken Sie nach Abschluss dieses Tutorials an Löschen des IAM-Benutzers. Für den Einsatz in der Produktion empfehlen wir, dass Sie die bewährten Sicherheitsmethoden in IAM im IAM-Benutzerhandbuch befolgen. Eine bewährte Methode ist, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden, um mit temporären Anmeldeinformationen auf AWS zuzugreifen. Eine weitere bewährte Methode besteht darin, dass Workloads temporäre Anmeldeinformationen mit IAM-Rollen verwenden müssen, um auf AWS zuzugreifen. Weitere Informationen zur Verwendung von AWS IAM Identity Center zum Erstellen von Benutzern mit temporären Anmeldeinformationen finden Sie unter Erste Schritte im AWS IAM Identity Center -Benutzerhandbuch.
> In diesem Tutorial werden auch Richtlinien mit vollem Zugriff verwendet. Für die Verwendung in der Produktion empfehlen wir, dass Sie stattdessen nur die für Ihren Anwendungsfall erforderlichen Mindestberechtigungen gemäß Bewährte Methoden in Bezug auf die Sicherheit erteilen.

Ihr IAM-Benutzer benötigt die folgenden AWS verwalteten Richtlinien:

- AmazonS3FullAccess – Gewährt Berechtigungen für alle Amazon S3-Aktionen, einschließlich Berechtigungen zum Erstellen und Verwenden eines Object Lambda Access Point.
- AWSLambda_FullAccess – Gewährt Berechtigungen für alle Lambda-Aktionen.
- AWSCloudFormationFullAccess – Gewährt Berechtigungen für alle AWS CloudFormation Aktionen.
- IAMFullAccess – Gewährt Berechtigungen für alle IAM-Aktionen.
- IAMAccessAnalyzerReadOnlyAccess – Gewährt Berechtigungen zum Lesen aller von IAM Access Analyzer bereitgestellten Zugriffsinformationen.

Sie können diese vorhandenen Richtlinien direkt anhängen, wenn Sie einen IAM-Benutzer erstellen. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen von IAM-Benutzern (Konsole) im IAM-Benutzerhandbuch.

Darüber hinaus erfordert Ihr IAM-Benutzer eine vom Kunden verwaltete Richtlinie. Um dem IAM-Benutzer Berechtigungen für alle AWS Serverless Application Repository Ressourcen und Aktionen zu erteilen, müssen Sie eine IAM-Richtlinie erstellen und die Richtlinie an den IAM-Benutzer anfügen.

Erstellen einer IAM-Richtlinie und Anfügen der Richtlinie an Ihren IAM-Benutzer

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich die Option Policies (Richtlinien) aus.
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie im Tab Visual editor (Visueller Editor) für Service Choose a service (Wählen Sie einen Service) aus. Wählen Sie dann Serverless Application Repository.
5. Wählen Sie für Aktionen unter Manuelle Aktionen Alle Aktionen des Serverless Application Repository (serverlessrepo: *) für dieses Tutorial.

   Als bewährte Methode für die Sicherheit sollten Sie basierend auf Ihrem Anwendungsfall nur für jene Aktionen und Ressourcen Berechtigungen zulassen, für die ein Benutzer Zugriff benötigt. Weitere Informationen finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.
6. Wählen Sie für Ressourcen Alle Ressourcen für dieses Tutorial.

   Als bewährte Methode sollten Sie Berechtigungen nur für bestimmte Ressourcen in bestimmten Konten definieren. Alternativ können Sie die geringste Berechtigung mit Bedingungsschlüsseln

erteilen. Weitere Informationen finden Sie unter Gewähren von geringsten Rechten im IAM-Benutzerhandbuch.

7. Wählen Sie Next: Tags (Weiter: Tags) aus.
8. Klicken Sie auf Weiter: Prüfen.
9. Geben Sie auf der Seite Review policy (Richtlinie überprüfen) einen Namen (z. B. **`tutorial-serverless-application-repository`**) und eine Beschreibung (optional) für die Richtlinie ein, die Sie erstellen. Überprüfen Sie die Richtlinienzusammenfassung, um sicherzustellen, dass Sie die beabsichtigten Berechtigungen erteilt haben, und wählen Sie dann Richtlinie erstellen aus, um Ihre neue Richtlinie zu speichern.
10. Wählen Sie im linken Navigationsbereich Benutzer aus. Wählen Sie dann den IAM-Benutzer für dieses Tutorial aus.
11. Wählen Sie auf der Seite Übersicht des ausgewählten Benutzers den Tab Berechtigungen und danach Hinzufügen von Berechtigungen.
12. Wählen Sie unter Berechtigungen gewähren die Option Vorhandene Richtlinien direkt anfügen aus.
13. Aktivieren Sie das Kontrollkästchen neben der gerade erstellten Richtlinie (z. B. **`tutorial-serverless-application-repository`**) und wählen Sie dann Nächstes: Überprüfung.
14. Prüfen Sie unter Richtlinienübersicht die Zusammenfassung, um sicherzustellen, dass Sie die beabsichtigte Richtlinie angefügt haben. Wählen Sie dann Add permissions (Berechtigungen hinzufügen) aus.

Schritt 1: Einen S3-Bucket erstellen

Erstellen Sie einen Bucket zum Speichern der ursprünglichen Daten, die Sie transformieren möchten.

So erstellen Sie einen Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Die Seite Create bucket (Bucket erstellen) wird geöffnet.
4. Geben Sie für Bucket-Name einen Namen für Ihren Bucket ein (z. B. **`tutorial-bucket`**).

Weitere Informationen zur Benennung von Buckets in Amazon S3 finden Sie unter Regeln für die Benennung von Buckets.

5. Unter Region wählen Sie die AWS-Region aus, in der sich der Bucket befinden soll.

   Weitere Informationen zur Bucket-Region finden Sie unter Bucket-Übersicht.

6. Belassen Sie die Einstellungen für den öffentlichen Zugriff für diesen Bucket blockieren bei den Standardeinstellungen (Alle öffentlichen Zugriffe blockieren ist aktiviert).

   Es wird empfohlen, alle Einstellungen für öffentlichen Zugriff blockieren aktiviert zu belassen, es sei denn, Sie müssen eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

7. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt.

   (Optional) Informationen zum Konfigurieren zusätzlicher Bucket-Einstellungen für Ihren spezifischen Anwendungsfall finden Sie unter Erstellen eines Buckets.

8. Wählen Sie Bucket erstellen aus.

Schritt 2: Hochladen einer Datei zu einem S3-Bucket

Laden Sie eine Textdatei mit bekannten PII-Daten verschiedener Typen, wie Namen, Bankinformationen, Telefonnummern und SSNs, in den S3-Bucket als Originaldaten hoch, von denen Sie PII später in diesem Tutorial bearbeiten werden.

Sie können beispielsweise nach der `tutorial.txt`-Datei hochladen. Dies ist ein Beispiel für eine Eingabedatei von Amazon Comprehend.

```
Hello Zhang Wei, I am John. Your AnyCompany Financial Services,
LLC credit card account 1111-0000-1111-0008 has a minimum payment
of $24.53 that is due by July 31st. Based on your autopay settings,
we will withdraw your payment on the due date from your
bank account number XXXXXX1111 with the routing number XXXXX0000.

Your latest statement was mailed to 100 Main Street, Any City,
WA 98121.
After your payment is received, you will receive a confirmation
text message at 206-555-0100.
If you have questions about your bill, AnyCompany Customer Service
```

```
is available by phone at 206-555-0199 or
email at support@anycompany.com.
```

Hochladen einer Datei in einen Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben, in den Sie die Datei hochladen möchten.
4. Wählen Sie im Tab Objects (Objekte) für Ihren Bucket die Option Upload (Hochladen) aus.
5. Wählen Sie auf der Seite Upload unter Dateien und Ordner die Option Dateien hinzufügen aus.
6. Wählen Sie eine hochzuladende Datei und dann Open (Öffnen) aus. Sie können z. B. das oben erwähnte `tutorial.txt`-Dateibeispiel hochladen.
7. Klicken Sie auf Hochladen.

Schritt 3: Erstellen eines S3-Zugriffspunkts

Um einen S3 Object Lambda Access Point für den Zugriff und die Umwandlung der ursprünglichen Daten zu verwenden, müssen Sie einen S3-Zugriffspunkt erstellen und dem S3-Bucket zuordnen, den Sie in Schritt 1 erstellt haben. Der Zugriffspunkt muss sich in derselben befinden AWS-Region wie die Objekte, die Sie transformieren möchten.

Später in diesem Tutorial verwenden Sie diesen Zugriffspunkt als unterstützenden Zugriffspunkt für Ihren Object Lambda Access Point.

So erstellen Sie einen Zugangspunkt

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffspunkte aus.
3. Wählen Sie auf der Seite Zugriffspunkte die Option Zugriffspunkt erstellen aus.
4. Geben Sie im Feld Name des Zugriffspunkts den gewünschten Namen für den Zugriffspunkt ein (z. B. **`tutorial-pii-access-point`**).

Weitere Hinweise zur Benennung von Zugangspunkten finden Sie unter Regeln zur Benennung von Amazon S3-Zugriffspunkten.

5. Wählen Sie im Feld Bucket-Name den Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben. S3 hängt den Zugriffspunkt an diesen Bucket an.

   (Optional) Sie können S3 durchsuchen auswählen, um die Buckets in Ihrem Konto zu browsen und zu durchsuchen. Wenn Sie Browse S3 (S3 durchsuchen) wählen, wählen Sie den gewünschten Bucket aus und dann Choose path (Pfad wählen), um das Feld Bucket name (Bucket-Name) mit dem Namen dieses Buckets zu füllen.

6. Wählen Sie für Netzwerkursprung Internetaus.

   Weitere Informationen zu Netzwerkursprüngen für Zugangspunkte finden Sie unter Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind.

7. Alle Block-Public-Access-Einstellungen sind standardmäßig für Zugriffspunkte aktiviert. Sie sollten alle Einstellungen für das Blockieren aller öffentlichen Zugriffe aktiviert lassen. Weitere Informationen finden Sie unter Verwalten des öffentlichen Zugriffs auf Zugriffspunkte.

8. Behalten Sie für alle anderen Zugriffspunktseinstellungen die Standardeinstellungen bei.

   (Optional) Sie können die Zugriffspunktseinstellungen ändern, um Ihren Anwendungsfall zu unterstützen. Für dieses Tutorial empfehlen wir, dass die Standardeinstellungen beibehalten werden.

   (Optional) Wenn Sie den Zugriff auf Ihren Zugriffspunkt verwalten müssen, können Sie eine Zugriffspunkt-Richtlinie angeben. Weitere Informationen finden Sie unter Beispiele von -Zugriffspunktrichtlinien.

9. Wählen Sie Zugangspunkt erstellen aus.

Schritt 4: Konfigurieren und Bereitstellen einer vordefinierten Lambda-Funktion

Um PII-Daten zu überarbeiten, konfigurieren und stellen Sie die vorgefertigte AWS Lambda -Funktion `ComprehendPiiRedactionS3ObjectLambda` für die Verwendung mit Ihrem S3 Object Lambda Access Point bereit.

Konfigurieren und Bereitstellen der Lambda-Funktion

1. Melden Sie sich bei der an AWS Management Console und zeigen Sie die `ComprehendPiiRedactionS3ObjectLambda` Funktion in der an AWS Serverless Application Repository.
2. Behalten Sie für Anwendungseinstellungen unter Anwendungsname den Standardwert (`ComprehendPiiRedactionS3ObjectLambda`) für dieses Tutorial bei.

   (Optional) Sie können den Namen eingeben, den Sie dieser Anwendung geben möchten. Sie können dies tun, wenn Sie planen, mehrere Lambda-Funktionen für unterschiedliche Zugriffsanforderungen für denselben freigegebenen Datensatz zu konfigurieren.
3. MaskCharacterBehalten Sie für den Standardwert (*****) bei. Das Maskenzeichen ersetzt jedes Zeichen in der geschärften PII-Entität.
4. MaskModeBehalten Sie für den Standardwert (MASK) bei. Der MaskMode Wert gibt an, ob die PII-Entität mit dem `MASK` Zeichen oder dem `PII_ENTITY_TYPE` Wert redigiert wird.
5. Um die angegebenen Datentypen zu redigieren, PiiEntityTypesbehalten Sie für den Standardwert ALL bei. Der PiiEntityTypes Wert gibt die PII-Entitätstypen an, die für die Schwärzung berücksichtigt werden sollen.

   Weitere Informationen zur Liste der unterstützten PII-Entitäts-Typen finden Sie unter Persönliche identifizierbare Informationen (PII) erkennen im Entwicklerhandbuch für Amazon Comprehend.
6. Lassen Sie die restlichen Einstellungen bei den Standardwerten.

   (Optional) Informationen zum Konfigurieren von zusätzlichen Einstellungen für Ihren speziellen Anwendungsfall finden Sie im Abschnitt Readme-Datei auf der linken Seite der Seite.
7. Aktivieren Sie das Kontrollkästchen neben Ich bestätige, dass diese App benutzerdefinierte IAM-Rollen erstellt.
8. Wählen Sie Bereitstellen.
9. Wählen Sie auf der Seite der neuen Anwendung unter Ressourcen dieLogische ID der Lambda-Funktion, die Sie bereitgestellt haben, um die Funktion auf der Lambda-Funktionsseite zu überprüfen.

Schritt 5: Erstellen eines S3 Object Lambda Access Point

Ein S3 Object Lambda Access Point bietet die Flexibilität, eine Lambda-Funktion direkt aus einer S3-GET-Anforderung aufzurufen, sodass die Funktion PII-Daten überarbeiten kann, die von einem S3-

Zugriffspunkt abgerufen wurden. Beim Erstellen und Konfigurieren eines S3 Object Lambda Access Point müssen Sie die überarbeitende Lambda-Funktion angeben, um den Ereigniskontext im JSON-Format als benutzerdefinierte Parameter für Lambda zu verwenden.

Der Ereigniskontext stellt Informationen über die Anforderung bereit, die in dem von S3 Object Lambda an Lambda übergebenen Ereignis gestellt wird. Weitere Informationen zu allen Feldern im Ereigniskontext finden Sie unter Format und Verwendung des Ereigniskontexts.

So erstellen Sie einen S3 Object Lambda Access Point

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie auf der Seite Object Lambda access points (Objekt-Lambda-Zugriffspunkte) die Option Create Object Lambda access point (Objekt-Lambda-Zugriffspunkt erstellen).
4. Geben Sie unter Name des Objekt-Lambda-Access Point den Namen ein, den Sie für den Object Lambda Access Point verwenden möchten (z. B. **`tutorial-pii-object-lambda-accesspoint`**).
5. Geben Sie für Unterstützenden Zugangspunkt den Standardzugangspunkt ein, den Sie in Schritt 3 erstellt haben (z. B. **`tutorial-pii-access-point`**), oder navigieren Sie zu diesem Zugangspunkt und wählen dann Unterstützenden Zugangspunkt auswählen aus.
6. Um für S3-APIs Objekte aus dem S3-Bucket abzurufen, die die Lambda-Funktion verarbeiten soll, wählen Sie ausGetObject.
7. Für die Funktion Lambda aufrufen können Sie eine der beiden folgenden Optionen für dieses Tutorial auswählen.
   - Wählen Sie Wählen Sie aus Funktionen in Ihrem Kontound wählen Sie die Lambda-Funktion aus, die Sie in Schritt 4 (z. B. **`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**) aus der Lambda-Funktion-Dropdown-Liste bereitgestellt haben.
   - Wählen Sie ARN eingeben und geben Sie dann den Amazon-Ressourcennamen (ARN) der Lambda-Funktion ein, die Sie in Schritt 4 erstellt haben.
8. Wählen Sie für Versioning der Lambda-Funktion, $LATEST (die neueste Version der Lambda-Funktion, die Sie in Schritt 4 erstellt haben).
9. (Optional) Wenn Sie Ihre Lambda-Funktion benötigen, um GET-Anforderungen mit Bereichs- und Teilenummern-Headern zu erkennen und zu bearbeiten, wählen Sie Lambda-Funktion

unterstützt Anforderungen mit Bereich und Lambda-Funktion unterstützt Anforderungen mit Teilenummern aus. Deaktivieren Sie andernfalls diese beiden Kontrollkästchen.

Weitere Informationen zur Verwendung von Bereichs- oder Teilenummern mit S3 Object Lambda finden Sie unter Arbeiten mit Range- und partNumber-Headern.

10. (Optional) Fügen Sie unter Nutzlast – optional JSON-Text hinzu, um Ihrer Lambda-Funktion zusätzliche Informationen bereitzustellen.

    Eine Nutzlast ist ein optionaler JSON-Text, den Sie Ihrer Lambda-Funktion als Eingabe für alle Aufrufe bereitstellen können, die von einem spezifischen S3 Object Lambda Access Point stammen. Um das Verhalten für mehrere Object Lambda Access Points anzupassen, die dieselbe Lambda-Funktion aufrufen, können Sie Nutzlasten mit unterschiedlichen Parametern konfigurieren und so die Flexibilität Ihrer Lambda-Funktion erweitern.

    Weitere Informationen zur Nutzlast finden Sie unter Format und Verwendung des Ereigniskontexts.

11. (Optional) Wählen Sie für Anforderungsmetriken – optional die Option Deaktivieren oder Aktivieren aus, um Ihrem Object Lambda Access Point Amazon-S3-Überwachung hinzuzufügen. Anforderungsmetriken werden zum Amazon- CloudWatch Standardtarif abgerechnet. Weitere Informationen finden Sie unter CloudWatch Preise.
12. Behalten Sie unter Objekt-Lambda-Zugriffspunkt-Richtlinie - optional die Standardeinstellung bei.

    (Optional) Sie können eine Ressourcenrichtlinie festlegen. Diese Ressourcenrichtlinie erteilt der `GetObject`-API die Berechtigung zur Verwendung des angegebenen Object Lambda Access Point.

13. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt und klicken Sie auf Erstellen eines Objekt-Lambda-Zugriffspunkts.

Schritt 6: Verwenden des S3 Object Lambda Access Point zum Abrufen der redigierten Daten

Jetzt ist S3 Object Lambda bereit, PII-Daten aus Ihrer Originaldatei zu redigieren.

So verwenden Sie den S3 Object Lambda Access Point zum Abrufen der redigierten Daten

Wenn Sie eine Datei über Ihren S3 Object Lambda Access Point abrufen möchten, führen Sie einen `GetObject`-API-Aufruf für S3 Object Lambda aus. S3 Object Lambda ruft die Lambda-Funktion auf,

um Ihre PII-Daten zu redigieren und gibt die transformierten Daten als Antwort auf die Standard-S3-`GetObject` API-Aufrufe.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie auf der Seite Objekt-Lambda-Zugriffspunkte den S3 Object Lambda Access Point aus, den Sie in Schritt 5 erstellt haben (z. B. **`tutorial-pii-object-lambda-accesspoint`**).
4. Wählen Sie auf der Registerkarte Objekte Ihres S3 Object Lambda Access Point die Datei mit dem gleichen Namen (z. B. `tutorial.txt`) wie diejenige aus, die Sie in Schritt 2 in den S3-Bucket hochgeladen haben.

   Diese Datei sollte alle transformierten Daten enthalten.
5. Um die transformierten Daten anzuzeigen, wählen Sie Öffnen oder Herunterladen aus.

   Sie sollten in der Lage sein, die redigierte Datei zu sehen, wie im folgenden Beispiel gezeigt.

```
Hello *********. Your AnyCompany Financial Services,
LLC credit card account ******************* has a minimum payment
of $24.53 that is due by *********. Based on your autopay settings,
we will withdraw your payment on the due date from your
bank account ********** with the routing number *********.

Your latest statement was mailed to *********************************.
After your payment is received, you will receive a confirmation
text message at ************.
If you have questions about your bill, AnyCompany Customer Service
is available by phone at ************ or
email at **********************.
```

Schritt 7: Bereinigen

Wenn Sie Ihre Daten nur zur Übung über S3 Object Lambda redigiert haben, löschen Sie die AWS Ressourcen, die Sie zugewiesen haben, damit keine Gebühren mehr anfallen.

Teilschritte

- Löschen des Object Lambda Access Point
- Löschen des S3-Zugriffspunkts
- Löschen Sie die Lambda-Funktion
- Löschen der CloudWatch Protokollgruppe
- Löschen Sie die Originaldatei im S3-Quell-Bucket
- Löschen des S3-Quell-Bucket
- Löschen der IAM-Rolle für Ihre Lambda-Funktion
- Löschen der vom Kunden verwalteten Richtlinie für Ihren IAM-Benutzer
- Löschen des IAM-Benutzers

Löschen des Object Lambda Access Point

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie auf der Seite Objekt-Lambda-Zugriffspunkte die Optionsschaltfläche links neben dem S3 Object Lambda Access Point aus, den Sie in Schritt 5 erstellt haben (z. B. **`tutorial-pii-object-lambda-accesspoint`**).
4. Wählen Sie Delete (Löschen).
5. Bestätigen Sie, dass Sie Ihren Object Lambda Access Point löschen möchten, indem Sie den Namen in das angezeigte Textfeld eingeben und dann Löschen wählen.

Löschen des S3-Zugriffspunkts

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffspunkte aus.
3. Navigieren Sie zu dem Zugriffspunkt, den Sie in Schritt 3 (z. B. **`tutorial-pii-access-point`**) erstellt haben, und wählen Sie die Optionsschaltfläche neben dem Namen des Zugriffspunkts aus.
4. Wählen Sie Löschen aus.
5. Bestätigen Sie, dass Sie Ihren Zugriffspunkt löschen möchten, indem Sie den Namen in das angezeigte Textfeld eingeben und dann Delete (Löschen) wählen.

Löschen Sie die Lambda-Funktion

1. Wählen Sie in der AWS Lambda -Konsole unter https://console.aws.amazon.com/lambda/ im linken Navigationsbereich Funktionen aus.
2. Wählen Sie die Funktion aus, die Sie in Schritt 4 (z. B. **`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**) erstellt haben.
3. Wählen Sie Aktionen und anschließend Löschen aus.
4. Wählen Sie im Bestätigungsdialogfeld Delete (Löschen) die Option Delete (Löschen) aus.

Löschen der CloudWatch Protokollgruppe

1. Öffnen Sie die - CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Log groups (Protokollgruppen) aus.
3. Suchen Sie die Protokollgruppe, deren Name mit der Lambda-Funktion endet, die Sie in Schritt 4 (z. B. **`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**) erstellt haben.
4. Wählen Sie Actions (Aktionen) und dann Delete log group(s) (Protokollgruppe(n) löschen) aus.
5. Wählen Sie im Dialogfeld Delete log group(s) (Protokollgruppe(n) löschen) die Option Delete (Löschen) aus.

Löschen Sie die Originaldatei im S3-Quell-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Bucket-Name den Namen des Buckets aus, in den Sie die Originaldatei in Schritt 2 (z. B. **`tutorial-bucket`**) hochgeladen haben.
4. Markieren Sie das Kontrollkästchen links neben dem Namen des Objekts, das Sie löschen möchten (z. B. `tutorial.txt`).
5. Wählen Sie Löschen aus.
6. Bestätigen Sie auf der Seite Löschen von Objekten im Abschnitt Objekte endgültig löschen?, dass Sie dieses Objekt löschen möchten, indem Sie **`permanently delete`** im Textfeld eingeben.
7. Wählen Sie Delete objects (Objekte löschen).

Löschen des S3-Quell-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets das Optionsfeld neben dem Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.
4. Wählen Sie Löschen aus.
5. Bestätigen Sie auf der Seite Delete bucket (Bucket löschen), dass Sie den Bucket löschen möchten. Geben Sie dazu den Bucket-Namen in das Textfeld ein und wählen Sie Delete bucket (Bucket löschen).

Löschen der IAM-Rolle für Ihre Lambda-Funktion

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles (Rollen), und aktivieren Sie dann das Kontrollkästchen neben der Rolle, die Sie löschen möchten. Der Rollenname beginnt mit dem Namen der Lambda-Funktion, die Sie in Schritt 4 (z. B. **`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**) erstellt haben.
3. Wählen Sie Löschen aus.
4. Geben Sie im Dialogfeld Löschen den Rollennamen in das Texteingabefeld ein, um das Löschen zu bestätigen. Wählen Sie dann Löschen aus.

Löschen der vom Kunden verwalteten Richtlinie für Ihren IAM-Benutzer

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich die Option Richtlinien aus.
3. Geben Sie auf der Seite Richtlinien den Namen der vom Kunden verwalteten Richtlinie ein, die Sie in Voraussetzungen (z. B. **`tutorial-serverless-application-repository`**) erstellt haben, in das Suchfeld ein, um die Richtlinienliste zu filtern. Aktivieren Sie die Optionsschaltfläche neben dem Namen der Richtlinie, die Sie löschen möchten.
4. Wählen Sie Aktionen und anschließend Löschen aus.

5. Bestätigen Sie, dass Sie diese Richtlinie löschen möchten, indem Sie den Namen in das angezeigte Textfeld eingeben und dann Delete (Löschen) wählen.

Löschen des IAM-Benutzers

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Benutzer, und aktivieren Sie dann das Kontrollkästchen neben dem Benutzer, den Sie löschen möchten.
3. Wählen Sie oben auf der Seite Delete (Löschen) aus.
4. Geben Sie im Dialogfeld ***Benutzernamen*** löschen? den Benutzernamen in das Texteingabefeld ein, um das Löschen des Benutzers zu bestätigen. Wählen Sie Delete (Löschen) aus.

Nächste Schritte

Nachdem Sie dieses Tutorial abgeschlossen haben, können Sie die folgenden verwandten Anwendungsfälle näher untersuchen:

- Sie können mehrere S3 Object Lambda Access Points erstellen und diese mit vorgefertigten Lambda-Funktionen aktivieren, die unterschiedlich konfiguriert sind, um bestimmte Typen von PII je nach den geschäftlichen Anforderungen der Datenzugreifer zu bearbeiten.

  Jeder Benutzertyp nimmt eine IAM-Rolle an und hat nur auf einen S3 Object Lambda Access Point Zugriff (verwaltet durch IAM-Richtlinien). Dann fügen Sie jede `ComprehendPiiRedactionS3ObjectLambda`-Lambda-Funktion, die für einen anderen Redigierungsanwendungsfall konfiguriert ist, an einen anderen S3 Object Lambda Access Point an. Für jeden S3 Object Lambda Access Point können Sie über einen unterstützenden S3-Zugriffspunkt verfügen, um Daten aus einem S3-Bucket zu lesen, der den freigegebenen Datensatz speichert.

  Weitere Informationen zum Erstellen einer S3-Bucket-Richtlinie, die es Benutzern ermöglicht, nur über S3-Zugriffspunkte aus dem Bucket zu lesen, finden Sie unter Konfigurieren von IAM-Richtlinien für die Verwendung von Zugriffspunkten.

  Weitere Informationen dazu, wie Sie einem Benutzer die Berechtigung für den Zugriff auf die Lambda-Funktion, den S3-Zugriffspunkt und den S3 Object Lambda Access Point erteilen, finden Sie unter Konfigurieren von IAM-Richtlinien für Object Lambda Access Points.

- Sie können Ihre eigene Lambda-Funktion erstellen und S3 Object Lambda mit Ihrer angepassten Lambda-Funktion verwenden, um Ihre spezifischen Datenanforderungen zu erfüllen.

  Um beispielsweise verschiedene Datenwerte zu untersuchen, können Sie S3 Object Lambda und Ihre eigene Lambda-Funktion verwenden, die zusätzliche Amazon-Comprehend-Funktionen verwenden, z. B. Entitätserkennung, Schlüsselphrasenerkennung, Stimmungsanalyse und Dokumentklassifizierung, um Daten zu verarbeiten. Sie können S3 Object Lambda auch zusammen mit Amazon Comprehend Medical, einem HIPAA-berechtigten NLP-Dienst, nutzen, um Daten kontextbewusst zu analysieren und zu extrahieren.

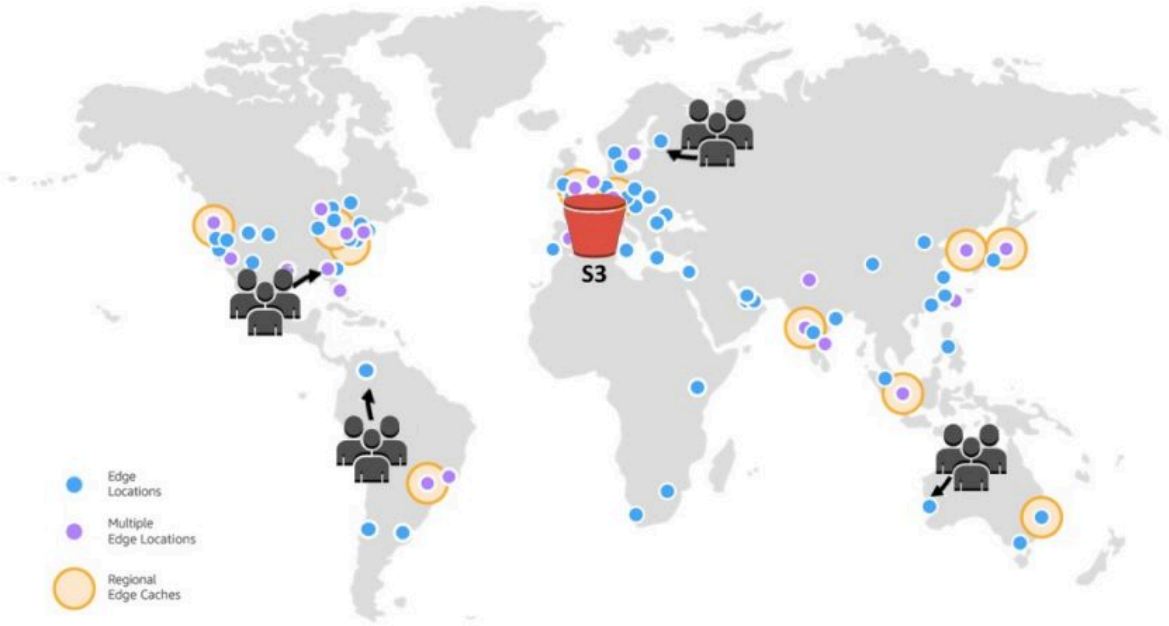
  Weitere Informationen zum Transformieren von Daten mit S3 Object Lambda und Ihrer eigenen Lambda-Funktion finden Sie unter Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda.

Tutorial: Hosten von On-Demand-Streaming-Videos mit Amazon S3 CloudFront, Amazon und Amazon Route 53

Sie können Amazon S3 mit Amazon verwenden CloudFront , um Videos für die On-Demand-Anzeige auf sichere und skalierbare Weise zu hosten. Für das Video-on-Demand-Streaming (VOD-Streaming) werden Ihre Videoinhalte auf einem Server gespeichert und Viewer können ihn jederzeit angezeigt werden.

CloudFront ist ein schneller, hochsicherer und programmierbarer Content Delivery Network (CDN)-Service. CloudFront kann Ihre Inhalte sicher über HTTPS von allen CloudFront Edge-Standorten auf der ganzen Welt aus bereitstellen. Weitere Informationen zu CloudFrontfinden Sie unter Was ist Amazon CloudFront? im Amazon- CloudFront Entwicklerhandbuch.

CloudFront Das Caching reduziert die Anzahl der Anfragen, auf die Ihr Ursprungsserver direkt reagieren muss. Wenn ein Betrachter (Endbenutzer) ein Video anfordert, das Sie mit bereitstellen CloudFront, wird die Anforderung an einen nahe gelegenen Edge-Standort weitergeleitet, der näher an dem Standort liegt, an dem sich der Betrachter befindet. CloudFront stellt das Video aus seinem Cache bereit und ruft es nur dann aus dem S3-Bucket ab, wenn es noch nicht zwischengespeichert ist. Diese Caching-Verwaltungsfunktion beschleunigt die Bereitstellung Ihres Videos für Viewer weltweit mit geringer Latenz, hohem Durchsatz und hohen Übertragungsgeschwindigkeiten. Weitere Informationen zur CloudFront Caching-Verwaltung finden Sie unter Optimieren von Caching und Verfügbarkeit im Amazon- CloudFront Entwicklerhandbuch.



Ziel

In diesem Tutorial konfigurieren Sie einen S3-Bucket für das Hosten von On-Demand-Video-Streaming mit CloudFront für die Bereitstellung und Amazon Route 53 für Domain Name System (DNS) und benutzerdefinierte Domänenverwaltung.

Themen

- (Optional) Schritt 8: Anzeigen von Daten zu Anforderungen, die von Ihrer CloudFront Verteilung empfangen werden
- Schritt 9: Bereinigen
- Nächste Schritte

Voraussetzungen: Registrieren und Konfigurieren einer benutzerdefinierten Domäne mit Route 53

Bevor Sie mit diesem Tutorial beginnen, müssen Sie eine benutzerdefinierte Domäne (z. B. **`example.com`**) bei Route 53 registrieren und konfigurieren, damit Sie Ihre CloudFront Verteilung später für die Verwendung eines benutzerdefinierten Domänennamens konfigurieren können.

Ohne einen benutzerdefinierten Domänennamen ist Ihr S3-Video öffentlich zugänglich und über CloudFront unter einer URL gehostet, die der folgenden ähnelt:

```
https://CloudFront distribution domain name/Path to an S3 video
```

Beispiel: **`https://d111111abcdef8.cloudfront.net/sample.mp4`**

Nachdem Sie Ihre CloudFront Verteilung für die Verwendung eines benutzerdefinierten Domänennamens konfiguriert haben, der mit Route 53 konfiguriert ist, ist Ihr S3-Video öffentlich zugänglich und über CloudFront unter einer URL gehostet, die der folgenden ähnelt:

```
https://CloudFront distribution alternate domain name/Path to an S3 video
```

Beispiel: **`https://www.example.com/sample.mp4`** Ein benutzerdefinierter Domänenname ist für die Viewer einfacher und intuitiver zu verwenden.

Informationen zur Anmeldung einer benutzerdefinierten Domäne finden Sie unter Registrieren einer neuen Domäne mit Route 53 im Amazon Route 53-Entwicklerhandbuch.

Wenn Sie einen Domänennamen bei Route 53 registrieren, erstellt Route 53 die gehostete Zone für Sie, die Sie später in diesem Tutorial verwenden werden. In dieser gehosteten Zone speichern Sie Informationen darüber, wie Sie den Datenverkehr für Ihre Domäne weiterleiten, z. B. an eine Amazon EC2-Instance oder eine - CloudFront Verteilung.

Es fallen Gebühren für die Domainregistrierung, Ihre gehostete Zone und DNS-Anfragen an, die von Ihrer Domain empfangen werden. Weitere Informationen dazu finden Sie unter Amazon Route 53 – Preise.

> **Note**
>
> Wenn Sie eine Domäne registrieren, kostet diese sofort Geld. Dies kann nicht rückgängig gemacht werden. Sie können wählen, die Domäne nicht automatisch zu erneuern, aber Sie bezahlen im Voraus und besitzen sie für das Jahr. Weitere Informationen finden Sie unter Registrieren einer neuen Domäne im Amazon-Route-53-Entwicklerhandbuch.

Schritt 1: Einen S3-Bucket erstellen

Erstellen Sie einen Bucket, um das Originalvideo zu speichern, das Sie streamen möchten.

So erstellen Sie einen Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Die Seite Bucket erstellen wird geöffnet.
4. Geben Sie im Feld Bucket Name einen Namen für Ihren Bucket ein (z. B. **`tutorial-bucket`**).

   Weitere Informationen zur Benennung von Buckets in Amazon S3 finden Sie unter Regeln für die Benennung von Buckets.
5. Wählen Sie für Region die aus, AWS-Region in der sich der Bucket befinden soll.

   Wenn möglich, sollten Sie die Region auswählen, die der Mehrheit Ihrer Viewer am nächsten ist. Weitere Informationen zur Bucket-Region finden Sie unter Bucket-Übersicht.
6. Belassen Sie die Einstellungen für den öffentlichen Zugriff für diesen Bucket blockieren bei den Standardeinstellungen (Alle öffentlichen Zugriffe blockieren ist aktiviert).

   Auch wenn „Alle öffentlichen Zugriffe blockieren“ aktiviert ist, können Viewer weiterhin über auf das hochgeladene Video zugreifen CloudFront. Diese Funktion ist ein großer Vorteil der Verwendung von CloudFront zum Hosten eines in S3 gespeicherten Videos.

Es wird empfohlen, alle Einstellungen aktiviert zu belassen, es sei denn, Sie müssen eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

7. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt.

   (Optional) Informationen zum Konfigurieren zusätzlicher Bucket-Einstellungen für Ihren spezifischen Anwendungsfall finden Sie unter Erstellen eines Buckets.

8. Wählen Sie Bucket erstellen aus.

Schritt 2: Hochladen eines Videos in den S3-Bucket

Im folgenden Verfahren wird beschrieben, wie Sie eine Videodatei mithilfe der Konsole in einen S3-Bucket hochladen. Wenn Sie ein Video in S3 hochladen, können Sie auch Amazon S3 Transfer Acceleration nutzen, um schnelle und sichere Dateiübertragungen zu konfigurieren. Transfer Acceleration kann das Hochladen von Videos zu Ihrem S3-Bucket zur Übertragung größerer Videos über große Entfernungen beschleunigen. Weitere Informationen finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.

So laden Sie die Datei in einen Bucket hoch

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben, in den Sie die Datei hochladen möchten.
4. Wählen Sie im Tab Objects (Objekte) für Ihren Bucket die Option Upload (Hochladen) aus.
5. Wählen Sie auf der Seite Upload unter Dateien und Ordner die Option Dateien hinzufügen aus.
6. Wählen Sie eine hochzuladende Datei und dann Öffnen aus.

   Sie können beispielsweise eine Videodatei mit dem Namen `sample.mp4` hochladen.

7. Klicken Sie auf Hochladen.

Schritt 3: Erstellen einer CloudFront Ursprungszugriffsidentität

Um den direkten Zugriff auf das Video aus Ihrem S3-Bucket einzuschränken, erstellen Sie einen speziellen CloudFront Benutzer, der als Ursprungszugriffsidentität (OAI) bezeichnet wird. Sie verknüpfen die OAI zu Ihrer Distribution zu einem späteren Zeitpunkt in diesem Tutorial. Durch die Verwendung einer OAI stellen Sie sicher, dass Viewer das Video nicht umgehen CloudFront und direkt aus dem S3-Bucket abrufen können. Nur die CloudFront OAI kann auf die Datei im S3-Bucket zugreifen. Weitere Informationen finden Sie unter Beschränken des Zugriffs auf Amazon S3-Inhalte mithilfe einer OAI im Amazon- CloudFront Entwicklerhandbuch.

So erstellen Sie eine CloudFront OAI

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie im linken Navigationsbereich im Abschnitt Sicherheit Ursprungszugriff aus.
3. Wählen Sie unter der Registerkarte Identitäten Ursprungszugriffsidentität erstellen aus.
4. Geben Sie einen Namen ein (z. B. **`S3-OAI`**) als die neue Ursprungszugriffsidentität ein.
5. Wählen Sie Erstellen aus.

Schritt 4: Erstellen einer CloudFront Verteilung

Um CloudFront zum Bereitstellen und Verteilen des Videos in Ihrem S3-Bucket zu verwenden, müssen Sie eine CloudFront Verteilung erstellen.

Teilschritte

- Erstellen einer CloudFront Verteilung
- Überprüfen der Bucket-Richtlinie

Erstellen einer CloudFront Verteilung

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie im linken Navigationsbereich Verteilungen aus.
3. Wählen Sie Verteilung erstellen aus.

4. Wählen Sie im Abschnitt Ursprung für Ursprungsdomäne den Domänennamen Ihres S3-Ursprungs aus, der mit dem Namen des S3-Buckets beginnt, den Sie unter Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.
5. Wählen Sie für Ursprungszugriff die Option Legacy-Zugriffsidentitäten aus.
6. Wählen Sie unter Ursprungszugriffsidentität die Ursprungszugriffsidentität aus, die Sie in Schritt 3 (z. B. **`S3-OAI`**) erstellt haben.
7. Wählen Sie unter Bucket-Richtlinie die Option Ja, Bucket-Richtlinie aktualisieren aus.
8. Wählen Sie im Abschnitt Standard-Cacheverhalten unter Viewer-Protokollrichtlinie HTTP an HTTPS umleiten aus.

   Wenn die diese Funktion auswählen, werden HTTP-Anforderungen automatisch an HTTPS umgeleitet werden, um Ihre Website zu schützen und die Daten Ihrer Viewer zu schützen.
9. Behalten Sie für die anderen Einstellungen im Abschnitt Standard-Cacheverhalten die Standardwerte bei.

   (Optional) Sie können steuern, wie lange Ihre Datei in einem CloudFront Cache verbleibt, bevor eine weitere Anfrage an Ihren Ursprung CloudFront weiterleitet. Eine Reduzierung der Dauer ermöglicht Ihnen, dynamische Inhalte bereitzustellen. Eine Erhöhung der Dauer bedeutet, dass Ihre Viewer eine bessere Leistung erhalten, da es wahrscheinlicher ist, dass Ihre Dateien direkt vom Edge-Cache bereitgestellt werden. Eine längere Dauer verringert darüber hinaus die Last auf Ihrem Ursprung. Weitere Informationen finden Sie unter Verwalten, wie lange Inhalte im Cache bleiben (Ablauf) im Amazon- CloudFront Entwicklerhandbuch.
10. Behalten Sie für die anderen Abschnitte die restlichen Einstellungen auf die Standardwerte eingestellt.

    Weitere Informationen zu den verschiedenen Einstellungsoptionen finden Sie unter Werte, die Sie beim Erstellen oder Aktualisieren einer Verteilung angeben im Amazon- CloudFront Entwicklerhandbuch.
11. Klicken Sie unten auf der Seite auf Verteilung erstellen.
12. Auf der Registerkarte Allgemein für Ihre CloudFront Verteilung unter Details ändert sich der Wert der Spalte Letzte Änderung für Ihre Verteilung von Wird bereitgestellt in den Zeitstempel, zu dem die Verteilung zuletzt geändert wurde. Dieser Prozess dauert in der Regel einige Minuten.

Überprüfen der Bucket-Richtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zuvor als Ursprung Ihrer CloudFront Verteilung verwendet haben (z. B. **`tutorial-bucket`**).
4. Wählen Sie die Registerkarte Permissions (Berechtigungen).
5. Bestätigen Sie im Abschnitt Bucket-Richtlinie, dass eine Anweisung ähnlich der folgenden im Bucket-Richtlinientext angezeigt wird:

```
{
    "Version": "2008-10-17",
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity EH1HDMB1FH2TC"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::tutorial-bucket/*"
        }
    ]
}
```

Dies ist die Anweisung, die Ihre CloudFront Verteilung Ihrer Bucket-Richtlinie hinzugefügt hat, als Sie zuvor Ja, Bucket-Richtlinie aktualisieren ausgewählt haben.

Diese Aktualisierung der Bucket-Richtlinie zeigt an, dass Sie die CloudFront Verteilung erfolgreich konfiguriert haben, um den Zugriff auf den S3-Bucket einzuschränken. Aufgrund dieser Einschränkung kann auf Objekte im Bucket nur über Ihre CloudFront Verteilung zugegriffen werden.

Schritt 5: Zugreifen auf das Video über die CloudFront Verteilung

Jetzt CloudFront kann das in Ihrem S3-Bucket gespeicherte Video bereitstellen. Um über auf Ihr Video zuzugreifen CloudFront, müssen Sie Ihren CloudFront Verteilungsdomänennamen mit dem Pfad zum Video im S3-Bucket kombinieren.

So erstellen Sie eine URL zum S3-Video mithilfe des CloudFront Verteilungsdomänennamens

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie im linken Navigationsbereich Verteilungen aus.
3. Um den Namen der Verteilungsdomäne zu erhalten, gehen Sie wie folgt vor:

   a. Suchen Sie in der Spalte Ursprünge nach der richtigen CloudFront Verteilung, indem Sie nach ihrem Ursprungsnamen suchen, der mit dem S3-Bucket beginnt, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.

   b. Nachdem Sie die Verteilung in der Liste gefunden haben, erweitern Sie die Spalte Domainname, um den Wert des Domainnamens für Ihre CloudFront Verteilung zu kopieren.
4. Fügen Sie in einer neuen Browser-Registerkarte den kopierten Verteilungsdomänennamen ein.
5. Kehren Sie zum vorherigen Browser-Tab zurück und öffnen Sie die S3-Konsole unter https://console.aws.amazon.com/s3/.
6. Wählen Sie im linken Navigationsbereich Buckets aus.
7. Wählen Sie in der Liste der Buckets den Namen des Buckets, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.
8. Wählen Sie in der Liste Objekte den Namen des Videos aus, das Sie in Schritt 2 (z. B. `sample.mp4`) hochgeladen haben.
9. Kopieren Sie auf der Objektdetailseite im Übersicht über das Objekt den Wert des Schlüssel. Dieser Wert ist der Pfad zum hochgeladenen Videoobjekt im S3-Bucket.
10. Kehren Sie zur Browser-Registerkarte zurück, in die Sie zuvor den Namen der Verteilungsdomäne eingefügt haben und geben Sie einen Schrägstrich (**`/`**) nach dem Domainnamen der Distributionsdomäne ein und fügen Sie dann den Pfad zu dem Video, das Sie vorher kopiert haben, ein (zum Beispiel `sample.mp4`).

    Jetzt ist Ihr S3-Video öffentlich zugänglich und über CloudFront unter einer URL gehostet, die der folgenden ähnelt:

```
https://CloudFront distribution domain name/Path to the S3 video
```

Ersetzen Sie *CloudFront den Verteilungsdomänennamen* und *den Pfad zum S3-Video* durch die entsprechenden Werte. Eine Beispiel-URL lautet **https://d111111abcdef8.cloudfront.net/sample.mp4**.

Schritt 6: Konfigurieren Ihrer CloudFront Verteilung für die Verwendung Ihres benutzerdefinierten Domänennamens

Um Ihren eigenen Domänennamen anstelle des CloudFront Domänennamens in der URL für den Zugriff auf das S3-Video zu verwenden, fügen Sie Ihrer CloudFront Verteilung einen alternativen Domänennamen hinzu.

Teilschritte

- Ein SSL-Zertifikat anfordern
- Hinzufügen des alternativen Domänennamens zu Ihrer CloudFront Verteilung
- Erstellen Sie einen DNS-Datensatz, um den Datenverkehr von Ihrem alternativen Domänennamen an den Domänennamen Ihrer CloudFront Verteilung weiterzuleiten
- Überprüfen Sie, ob IPv6 für Ihre Verteilung aktiviert ist, und erstellen Sie bei Bedarf einen weiteren DNS-Eintrag

Ein SSL-Zertifikat anfordern

Damit Ihre Viewer HTTPS und Ihren benutzerdefinierten Domänennamen in der URL für Ihr Video-Streaming verwenden können, verwenden Sie AWS Certificate Manager (ACM), um ein Secure Sockets Layer (SSL)-Zertifikat anzufordern. Das SSL-Zertifikat stellt eine verschlüsselte Netzwerkverbindung zur Website her.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die ACM-Konsole unter https://console.aws.amazon.com/acm/.
2. Wenn die Einführungsseite angezeigt wird, klicken Sie unter Bereitstellung von Zertifikaten auf Erste Schritte.
3. Wählen Sie auf der Seite Zertifikat anfordern die Option Öffentliches Zertifikat anfordern und dann Zertifikat anfordern aus.

4. Geben Sie auf der Seite Domänennamen hinzufügen den vollqualifizierten Domänennamen (FQDN) der Website ein, die Sie mit einem SSL/TLS-Zertifikat sichern möchten. Sie können ein Sternchen (*) verwenden, um ein Platzhalterzertifikat anzufordern, um mehrere Websitenamen in derselben Domäne zu schützen. Geben Sie in diesem Tutorial ***** und den benutzerdefinierten Domänennamen ein, den Sie in Voraussetzungen konfiguriert haben. Verwenden Sie zum Beispiel ***.example.com** und wählen Sie dann Weiter aus.

   Weitere Informationen finden Sie unter So fordern Sie ein öffentliches ACM-Zertifikat an (Konsole) im AWS Certificate Manager -Benutzerhandbuch.
5. Klicken Sie auf der Seite Select validation method (Validierungsmethode auswählen) auf DNS validation (DNS-Validierung). Wählen Sie anschließend Weiter aus.

   Wenn Sie über die Berechtigung zum Ändern der DNS-Konfiguration verfügen, empfehlen wir, dass Sie die DNS-Domainvalidierung anstelle einer E-Mail-Validierung verwenden. Die DNS-Validierung hat mehrere Vorteile im Vergleich zur E-Mail-Validierung. Weitere Informationen finden Sie unter Option 1: DNS-Validierung im AWS Certificate Manager -Benutzerhandbuch.
6. (Optional) Auf der Seite Tags hinzufügen können Sie Ihr Zertifikat mit Metadaten markieren.
7. Wählen Sie Überprüfen aus.
8. Überprüfen Sie auf der Seite Prüfen, ob die Informationen unter Domänenname und Validierungsmethode korrekt sind. Wählen Sie Bestätigen und anfordern aus.

   Auf der Validierungs-Seite wird angezeigt, dass Ihre Anforderung verarbeitet wird und dass Zertifikatdomänen validiert werden. Das Zertifikat, das auf die Validierung wartet, befindet sich im Status Validierung ausstehend.
9. Wählen Sie auf der Validierungs-Seite den Abwärtspfeil links neben Ihrem benutzerdefinierten Domänennamen aus und wählen Sie dann Datensatz in Route 53 erstellen, um Ihren Domäneneigentümer über DNS zu überprüfen.

   Dadurch wird AWS Certificate Manager Ihrer DNS-Konfiguration ein von bereitgestellter CNAME-Datensatz hinzugefügt.
10. Wählen Sie im Dialogfeld Datensatz in Route 53 erstellen die Option Erstellen.

    Die Seite Validierung sollte unten eine Statusbenachrichtigung von Erfolg anzeigen.
11. Wählen Sie Continue (Fortfahren), um die Listenseite Certificates (Zertifikate) anzuzeigen.

    Der Status für Ihr neues Zertifikat wird sich innerhalb von 30 Minuten von Ausstehende Validierung auf Ausgestellt ändern.

Hinzufügen des alternativen Domänennamens zu Ihrer CloudFront Verteilung

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie im linken Navigationsbereich Verteilungen aus.
3. Wählen Sie die ID für die Verteilung aus, die Sie in Schritt 4 erstellt haben.
4. Gehen Sie auf der Registerkarte Allgemeines zum Abschnitt Einstellungen und wählen Sie Bearbeiten aus.
5. Wählen Sie auf der Seite Einstellungen bearbeiten für Alternativer Domänenname (CNAME) – optional die Option Element hinzufügen aus, um die benutzerdefinierten Domänennamen hinzuzufügen, die Sie in der URL für das von dieser CloudFront Verteilung bereitgestellte S3-Video verwenden möchten.

   Wenn Sie in diesem Tutorials beispielsweise Datenverkehr für eine Subdomäne weiterleiten möchten, z. B.`www.example.com` , geben Sie den Subdomänennamen (`www`) mit dem Domänennamen (`example.com`) ein. Geben Sie insbesondere **`www.example.com`** ein.

   > **Note**
   >
   > Der alternative Domänenname (CNAME), den Sie hinzufügen, muss durch das SSL-Zertifikat abgedeckt sein, das Sie zuvor an Ihre CloudFront Verteilung angehängt haben.

6. Wählen Sie für Benutzerdefiniertes SSL-Zertifikat – optional das vorher angeforderte SSL-Zertifikat aus (z. B. **`*.example.com`**).

   > **Note**
   >
   > Wenn das SSL-Zertifikat nicht unmittelbar nach der Anforderung angezeigt wird, warten Sie 30 Minuten und aktualisieren Sie die Liste, bis das SSL-Zertifikat ausgewählt werden kann.

7. Lassen Sie die restlichen Einstellungen bei den Standardwerten. Wählen Sie Save Changes (Änderungen speichern).
8. Warten Sie auf der Registerkarte Allgemeines für die Verteilung, bis der Wert Zuletzt geändert von Wird bereitgestellt in den Zeitstempel geändert wurde, als die Verteilung zuletzt geändert wurde.

Erstellen Sie einen DNS-Datensatz, um den Datenverkehr von Ihrem alternativen Domänennamen an den Domänennamen Ihrer CloudFront Verteilung weiterzuleiten

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Route-53-Konsole unter https://console.aws.amazon.com/route53/.
2. Wählen Sie im linken Navigationsbereich Hosted Zones (Bereitgestellte Zonen).
3. Wählen Sie auf der Seite Gehostete Zonen den Namen der gehosteten Zone aus, die Route 53 für Sie in Voraussetzungen erstellt hat (z. B. **`example.com`**).
4. Klicken Sie auf Datensatz erstellen und benutzen Sie dann die Datensatz Quick Create-Methode.
5. Behalten Sie für Datensatzname den Wert für den Datensatznamen mit dem alternativen Domänennamen der CloudFront Verteilung bei, die Sie zuvor hinzugefügt haben.

   Geben Sie in diesem Tutorial den Namen der Subdomäne ohne den Domänennamen ein, um Datenverkehr an eine Subdomäne weiterzuleiten, z. B. `www.example.com`. Geben Sie z. B. nur ein **`www`** im Textfeld vor Ihrem benutzerdefinierten Domänennamen ein.
6. Wählen Sie für Datensatztyp A – Leitet den Datenverkehr an eine IPv4-Adresse und einige AWS Ressourcen weiter.
7. Wählen Sie für Wert den Alias-Toggle, um die Alias-Ressource zu aktivieren.
8. Wählen Sie unter Datenverkehr an weiterleiten aus der Dropdown-Liste Alias zur CloudFront Verteilung aus.
9. Wählen Sie im Suchfeld mit der Bezeichnung Verteilung auswählen den Domänennamen der CloudFront Verteilung aus, die Sie in Schritt 4 erstellt haben.

   Gehen Sie wie folgt vor, um den Domänennamen Ihrer CloudFront Verteilung zu finden:

   a. Melden Sie sich in einer neuen Browser-Registerkarte bei der AWS Management Console an und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v3/home.
   b. Wählen Sie im linken Navigationsbereich Verteilungen aus.
   c. Suchen Sie in der Spalte Ursprünge nach der richtigen CloudFront Verteilung, indem Sie nach ihrem Ursprungsnamen suchen, der mit dem S3-Bucket beginnt, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.
   d. Nachdem Sie die Verteilung in der Liste gefunden haben, erweitern Sie die Spalte Domänenname, um den Domänennamenwert für Ihre CloudFront Verteilung anzuzeigen.

10. Behalten Sie auf der Seite Datensatz erstellen in der Route 53-Konsole die Standardwerte für die Übrigen Einstellungen bei.
11. Wählen Sie Create records (Datensätze erstellen).

Überprüfen Sie, ob IPv6 für Ihre Verteilung aktiviert ist, und erstellen Sie bei Bedarf einen weiteren DNS-Eintrag

Wenn IPv6 für Ihre Verteilung aktiviert ist, müssen Sie einen weiteren DNS-Eintrag erstellen.

1. Gehen Sie wie folgt vor, um zu überprüfen, ob IPv6 für Ihre Verteilung aktiviert ist:

    a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.

    b. Wählen Sie im linken Navigationsbereich Verteilungen aus.

    c. Wählen Sie die ID der CloudFront Verteilung aus, die Sie in Schritt 4 erstellt haben.

    d. Überprüfen Sie auf der Registerkarte Allgemeines unter Einstellungen, obIPv6 auf Aktiviert gesetzt ist.

    Wenn IPv6 für Ihre Verteilung aktiviert ist, müssen Sie einen weiteren DNS-Eintrag erstellen.
2. Wenn IPv6 für Ihre Verteilung aktiviert ist, gehen Sie wie folgt vor, um einen DNS-Eintrag zu erstellen:

    a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Route-53-Konsole unter https://console.aws.amazon.com/route53/.

    b. Wählen Sie im linken Navigationsbereich Hosted Zones (Bereitgestellte Zonen).

    c. Wählen Sie auf der Seite Gehostete Zonen den Namen der gehosteten Zone aus, die Route 53 für Sie in Voraussetzungen erstellt hat (z. B. **`example.com`**).

    d. Klicken Sie auf Datensatz erstellen und benutzen Sie dann die Datensatz Quick Create-Methode.

    e. Geben Sie für Datensatzname im Textfeld vor Ihrem benutzerdefinierten Domänennamen den gleichen Wert ein, den Sie beim Erstellen des IPv4-DNS-Eintrags vorher eingegeben haben. Geben Sie in diesem Tutorial beispielsweise nur **`www`** ein, um den Datenverkehr für die Subdomäne `www.example.com` weiterzuleiten.

    f. Wählen Sie für Record type (Datensatztyp) die Option AAAA – Routes traffic to an IPv6 address and some AWS ressources (Leitet Datenverkehr an eine IPv6-Adresse und einige AWS -Ressourcen weiter).

g. Wählen Sie für Wert den Alias-Toggle, um die Alias-Ressource zu aktivieren.

h. Wählen Sie unter Datenverkehr an weiterleiten aus der Dropdown-Liste Alias zur CloudFront Verteilung aus.

i. Wählen Sie im Suchfeld mit der Bezeichnung Verteilung auswählen den Domänennamen der CloudFront Verteilung aus, die Sie in Schritt 4 erstellt haben.

j. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt.

k. Wählen Sie Create records (Datensätze erstellen).

Schritt 7: Zugreifen auf das S3-Video über die CloudFront Verteilung mit dem benutzerdefinierten Domänennamen

Um mithilfe der benutzerdefinierten URL auf das S3-Video zuzugreifen, müssen Sie Ihren alternativen Domänennamen mit dem Pfad zum Video im S3-Bucket kombinieren.

So erstellen Sie eine benutzerdefinierte URL für den Zugriff auf das S3-Video über die CloudFront Verteilung

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie im linken Navigationsbereich Verteilungen aus.
3. Gehen Sie wie folgt vor, um den alternativen Domänennamen Ihrer CloudFront Verteilung abzurufen:

   a. Suchen Sie in der Spalte Ursprünge nach der richtigen CloudFront Verteilung, indem Sie nach ihrem Ursprungsnamen suchen, der mit dem S3-Bucket-Namen für den Bucket beginnt, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.

   b. Nachdem Sie die Verteilung in der Liste gefunden haben, erweitern Sie die Spalte Alternative Domänennamen, um den Wert des alternativen Domänennamens Ihrer CloudFront Verteilung zu kopieren.
4. Fügen Sie in einer neuen Browser-Registerkarte den alternativen Domänennamen der CloudFront Verteilung ein.
5. Kehren Sie zur vorherigen Browser-Registerkarte zurück, und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
6. Finden Sie den Pfad zu Ihrem S3-Video, wie im Schritt 5 erklärt wird.

7. Kehren Sie zur Browserregisterkarte zurück, auf der Sie zuvor den alternativen Domänennamen eingefügt haben, geben Sie einen Vorwärts-Schrägstrich (**/**) ein und fügen Sie dann den Pfad in Ihr S3-Video ein (z. B. `sample.mp4`).

   Jetzt ist Ihr S3-Video öffentlich zugänglich und über CloudFront unter einer benutzerdefinierten URL gehostet, die der folgenden ähnelt:

   ```
   https://CloudFront distribution alternate domain name/Path to the S3 video
   ```

   Ersetzen Sie den *CloudFront alternativen Domänennamen der Verteilung* und *den Pfad zum S3-Video* durch die entsprechenden Werte. Eine Beispiel-URL lautet **https://www.example.com/sample.mp4**.

(Optional) Schritt 8: Anzeigen von Daten zu Anforderungen, die von Ihrer CloudFront Verteilung empfangen werden

So zeigen Sie Daten zu Anforderungen an, die von Ihrer CloudFront Verteilung empfangen wurden

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie im linken Navigationsbereich unter Berichte & Analytik die Berichte aus der Konsole aus, die sich über Cache-Statistiken, Beliebte Objekte, Top-Referrer, Verwendung bis zu Viewer erstrecken.

   Sie können jedes Berichts-Dashboard filtern. Weitere Informationen finden Sie unter CloudFront Berichte in der -Konsole im Amazon- CloudFront Entwicklerhandbuch.
3. Um Daten zu filtern, wählen Sie die ID der CloudFront Verteilung aus, die Sie in Schritt 4 erstellt haben.

Schritt 9: Bereinigen

Wenn Sie ein S3-Streaming-Video mit CloudFront und Route 53 nur als Lernübung gehostet haben, löschen Sie die AWS Ressourcen, die Sie zugewiesen haben, damit keine Gebühren mehr anfallen.

> **Note**
>
> Wenn Sie eine Domäne registrieren, kostet diese sofort Geld. Dies kann nicht rückgängig gemacht werden. Sie können wählen, die Domäne nicht automatisch zu erneuern, aber Sie bezahlen im Voraus und besitzen sie für das Jahr. Weitere Informationen finden Sie unter Registrieren einer neuen Domäne im Amazon-Route-53-Entwicklerhandbuch.

Teilschritte

- Löschen der CloudFront Verteilung
- Löschen Sie den DNS-Datensatz
- Löschen Sie die öffentlich gehostete Zone für Ihre benutzerdefinierte Domäne
- Löschen des benutzerdefinierten Domänennamens aus Route 53
- Löschen des Originalvideos im S3-Quell-Bucket
- Löschen des S3-Quell-Bucket

Löschen der CloudFront Verteilung

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie im linken Navigationsbereich Verteilungen aus.
3. Suchen Sie in der Spalte Ursprünge nach der richtigen CloudFront Verteilung, indem Sie nach ihrem Ursprungsnamen suchen, der mit dem S3-Bucket-Namen für den Bucket beginnt, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.
4. Um die CloudFront Verteilung zu löschen, müssen Sie sie zuerst deaktivieren.
   - Wenn der Wert der Spalte Status Aktiviert ist und der Wert von Letzte Änderung der Zeitstempel ist, in dem die Verteilung zuletzt geändert wurde, deaktivieren Sie die Verteilung weiterhin, bevor Sie sie löschen.
   - Wenn der Wert von Status Aktiviert ist und der Wert von Letzte Änderung Wird bereitgestellt ist, warten Sie, bis sich der Wert von Status in den Zeitstempel ändert, als die Verteilung zuletzt geändert wurde. Fahren Sie dann fort, um die Verteilung vor dem Löschen zu deaktivieren.
5. Gehen Sie wie folgt vor, um die CloudFront Verteilung zu deaktivieren:

a. Wählen Sie in der Verteilungen-Liste das Kontrollkästchen neben der ID für die Verteilung, die Sie löschen möchten.

b. Um die Verteilung zu deaktivieren, wählen Sie Deaktivieren, und wählen Sie dann Deaktivieren, um zu bestätigen.

   Wenn Sie eine Verteilung deaktivieren, der ein alternativer Domänenname zugeordnet ist, CloudFront stoppt die Annahme von Datenverkehr für diesen Domänennamen (z. B. `www.example.com`), auch wenn eine andere Verteilung einen alternativen Domänennamen mit einem Platzhalter (`*`) hat, der derselben Domäne entspricht (z. B. `*.example.com`).

c. Der Wert von State (Zustand) wird sofort in Disabled (Deaktiviert) geändert. Warten Sie bis der Wert Zuletzt geändert von Wird bereitgestellt in den Zeitstempel geändert wurde, als die Verteilung zuletzt geändert wurde.

   Da diese Änderung an alle Edge-Standorte weitergeben CloudFront muss, kann es einige Minuten dauern, bis die Aktualisierung abgeschlossen ist und die Option Löschen verfügbar ist, damit Sie die Verteilung löschen können.

6. Gehen Sie wie folgt vor, um die deaktivierte Verteilung zu löschen:

   a. Aktivieren Sie das Kontrollkästchen neben der ID für die Verteilung, die Sie löschen möchten.

   b. Wählen Sie Löschen und Löschen aus, um den Vorgang zu bestätigen.

Löschen Sie den DNS-Datensatz

Wenn Sie die öffentliche gehostete Zone für die Domain (einschließlich des DNS-Datensatzes) löschen möchten, lesen Sie Löschen Sie die öffentlich gehostete Zone für Ihre benutzerdefinierte Domäne im Entwicklerhandbuch für Amazon Route 53. Wenn Sie nur den DNS-Eintrag löschen möchten, der in Schritt 6 erstellt wurde, tun Sie Folgendes:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Route-53-Konsole unter https://console.aws.amazon.com/route53/.
2. Wählen Sie im linken Navigationsbereich Hosted Zones (Bereitgestellte Zonen).
3. Wählen Sie auf der Seite Gehostete Zonen den Namen der gehosteten Zone aus, die Route 53 für Sie in Voraussetzungen erstellt hat (z. B. **`example.com`**).
4. Wählen Sie in der Liste der Datensätze das Kontrollkästchen neben den Datensätzen aus, die Sie löschen möchten (die Datensätze, die Sie in Schritt 6 erstellt haben).

> **Note**
>
> Sie können keine Datensätze löschen, die über einen Typen-Wert von NS oder SOA verfügen.

5. Wählen Sie Delete Records (Datensätze löschen).
6. Um die Löschung zu bestätigen, klicken Sie auf Delete (Löschen).

Änderungen an Datensätzen nehmen etwas Zeit in Anspruch, um auf die Route 53-DNS-Server weitergegeben zu werden. Derzeit besteht die einzige Möglichkeit, zu überprüfen, ob Ihre Änderungen weitergegeben wurden, darin, die GetChange API-Aktion zu verwenden. Änderungen werden im Allgemeinen innerhalb von 60 Sekunden an alle Route 53-Name-Server übertragen.

Löschen Sie die öffentlich gehostete Zone für Ihre benutzerdefinierte Domäne

> **Warning**
>
> Wenn Sie Ihre Domänenregistrierung behalten möchten, aber die Weiterleitung von Internetdatenverkehr an Ihre Website oder Webanwendung beenden möchten, empfehlen wir, dass Sie Datensätze in der gehosteten Zone (wie im obigen Abschnitt beschrieben) löschen, statt die gehostete Zone zu löschen.
> Wenn Sie eine gehostete Zone löschen, könnte jemand anderes die Domäne nutzen und den Datenverkehr über Ihren Domänennamen an die eigenen Ressourcen weiterleiten.
> Wenn Sie eine gehostete Zone löschen, können Sie diese außerdem nicht wiederherstellen. Sie müssen eine neue gehostete Zone erstellen und Sie die Namensserver für Ihre Domain-Registrierung aktualisieren. Es kann bis zu 48 Stunden dauern, bis diese Einstellungen wirksam werden.
> Wenn Sie möchten, dass die Domäne im Internet nicht verfügbar ist, können Sie Ihren DNS-Service auf einen kostenlosen DNS-Service übertragen und dann die in Route 53-gehostete-Zone löschen. Dadurch wird verhindert, dass zukünftige DNS-Abfragen möglicherweise fehlgeleitet werden.

1. Wenn die Domäne bei Route 53 registriert wurde, lesen Sie Hinzufügen oder Ändern der Namensserver und Verbindungsdatensätze in einer Domäne im Amazon Route 53-

Entwicklerhandbuch für Informationen darüber, wie Sie Route 53-Namensserver durch Namensserver für den neuen DNS-Service ersetzen können.

2. Wenn die Domäne mit einer anderen Vergabestellt registriert wurde, verwenden Sie die Methode der Vergabestelle, um Namensserver für die Domäne zu ändern.

> **Note**
>
> Wenn Sie eine gehostete Zone für eine Subdomäne (`www.example.com`) löschen, müssen Sie keine Namensserver für die Domäne (`example.com`) ändern.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Route-53-Konsole unter https://console.aws.amazon.com/route53/.
2. Wählen Sie im linken Navigationsbereich Hosted Zones (Bereitgestellte Zonen).
3. Wählen Sie auf der Seite Hosted Zones den Namen für die gehostete Zone aus, die Sie löschen möchten.
4. Vergewissern Sie sich im Tab Records (Datensätze) für Ihre gehostete Zone, dass die gehostete Zone, die Sie löschen möchten, nur einen NS- und einen SOA-Datensatz enthält.

   Enthält sie zusätzliche Datensätze, löschen Sie diese zuerst.

   Wenn Sie NS-Datensätze für Subdomänen in der gehosteten Zone erstellt haben, löschen Sie auch diese Datensätze.
5. Deaktivieren Sie im Tab DNSSEC signing (DNSSEC-Signierung) für Ihre gehostete Zone die DNNSSEC-Signierung, wenn sie aktiviert wurde. Weitere Informationen finden Sie unter Deaktivieren von DNS-Signaturen im Entwicklerhandbuch für Amazon Route 53.
6. Wählen Sie oben auf der Detailseite der gehosteten Zone Zone löschen aus.
7. Geben Sie **`delete`** ein, um das Löschen zu bestätigen, und wählen Sie dann Löschen.

Löschen des benutzerdefinierten Domänennamens aus Route 53

Für die meisten Domains oberster Ebene (Top-Level-Domains, TLDs) können Sie die Registrierung löschen, wenn Sie sie nicht mehr benötigen. Wenn Sie eine Domänennamenregistrierung aus Route 53 löschen, bevor die Registrierung abläuft, übernimmt die Registrierungsgebühr AWS nicht. Weitere

Informationen finden Sie unter Löschen einer Domainnamenregistrierung im Amazon-Route 53-Entwicklerhandbuch.

> **Important**
>
> Wenn Sie die Domain zwischen AWS-Konten oder an eine andere Vergabestelle übertragen möchten, löschen Sie die Domain nicht und erwarten Sie, sie sofort erneut zu registrieren. Lesen Sie stattdessen die entsprechende Dokumentation im Entwicklerhandbuch für Amazon Route 53:
>
> - Übertragen einer Domäne an ein anderes AWS-Konto
> - Überträgt eine Domain von Amazon Route 53 zu einer anderen Vergabestelle

Löschen des Originalvideos im S3-Quell-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Bucket-Name den Namen des Buckets aus, in den Sie das Video inSchritt 2 (z. B. **`tutorial-bucket`**) hochgeladen haben.
4. Markieren Sie auf der Registerkarte Objekte das Kontrollkästchen neben dem Namen des Objekts, das Sie löschen möchten (z. B. `sample.mp4`).
5. Wählen Sie Löschen aus.
6. Geben Sie unter Objekte endgültig löschen? den Wert **`permanently delete`** um zu bestätigen, dass Sie dieses Objekt löschen möchten.
7. Wählen Sie Delete objects (Objekte löschen).

Löschen des S3-Quell-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets das Optionsfeld neben dem Namen des Buckets aus, den Sie in Schritt 1 (z. B. **`tutorial-bucket`**) erstellt haben.

4. Wählen Sie Löschen aus.
5. Bestätigen Sie auf der Seite Delete bucket (Bucket löschen), dass Sie den Bucket löschen möchten. Geben Sie dazu den Bucket-Namen in das Textfeld ein und wählen Sie Delete bucket (Bucket löschen).

Nächste Schritte

Nachdem Sie dieses Tutorial abgeschlossen haben, können Sie die folgenden verwandten Anwendungsfälle näher untersuchen:

- Transcodieren Sie S3-Videos in Streaming-Formate, die von einem bestimmten Telefonie- oder angeschlossenen Gerät benötigt werden, bevor diese Videos mit einer - CloudFront Verteilung gehostet werden.

  Informationen zur Verwendung von Amazon S3 Batch Operations AWS Lambda und AWS Elemental MediaConvert zur Batch-Transcodierung einer Sammlung von Videos in eine Vielzahl von Ausgabe-Medienformaten finden Sie unter Tutorial: Batch-Transcodierung von Videos mit S3-AWS Lambda Batchoperationen und AWS Elemental MediaConvert.
- Hosten Sie andere in S3 gespeicherte Objekte wie Bilder, Audio, Bewegungsgrafiken, Stylesheets, HTML, JavaScriptReact-Apps usw. mit CloudFront und Route 53.

  Ein Beispiel finden Sie unter Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne und Beschleunigen Ihrer Website mit Amazon CloudFront.
- Verwenden Sie Amazon S3 Transfer Acceleration, um schnelle und sichere Dateiübertragungen zu konfigurieren. Transfer Acceleration kann das Hochladen von Videos zu Ihrem S3-Bucket zur Übertragung größerer Videos über große Entfernungen beschleunigen. Transfer Acceleration verbessert die Übertragungsleistung, indem der Datenverkehr über die CloudFront global verteilten Edge-Standorte und über die AWS Backbone-Netzwerke geleitet wird. Es verwendet auch Netzwerkprotokoll-Optimierungen. Weitere Informationen finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.

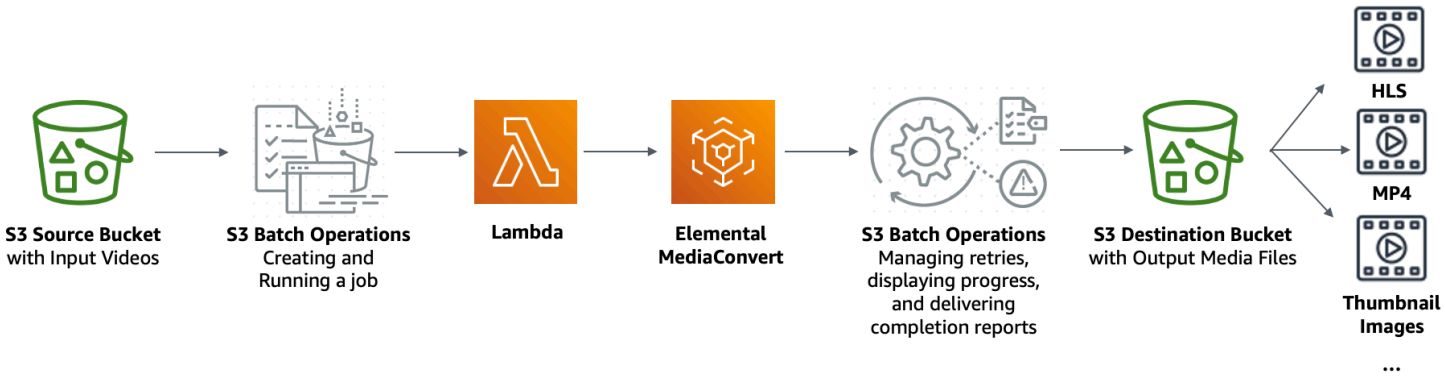
Tutorial: Batch-Transcodierung von Videos mit S3- AWS Lambda Batchoperationen und AWS Elemental MediaConvert

Videokonsumenten nutzen Geräte aller Formen, Größen und Jahrgänge, um Medieninhalte zu genießen. Diese breite Palette an Geräten stellt eine Herausforderung für Content-Ersteller und -Distributoren dar. Anstatt in einem one-size-fits-all Format zu sein, müssen Videos konvertiert werden, damit sie eine breite Palette von Größen, Formaten und Bitraten umfassen können. Diese Konvertierungsaufgabe ist noch schwieriger, wenn Sie eine große Anzahl von Videos haben, die konvertiert werden müssen.

AWS bietet Ihnen eine Methode zum Erstellen einer skalierbaren, verteilten Architektur, die Folgendes tut:

- Greift Eingabevideos auf
- Verarbeitet die Videos zur Wiedergabe auf einer Vielzahl von Geräten
- Speichert die transcodierten Mediendateien
- Liefert die Ausgabemediendateien und befriedigt Ansprüche

Wenn Sie umfangreiche Video-Repositories in Amazon S3 gespeichert haben, können Sie diese Videos aus ihren Quellformaten in mehrere Dateitypen in der Größe, Auflösung und Format transkodieren, die von einem bestimmten Videoplayer oder Gerät benötigt werden. Insbesondere bietet Ihnen S3-Batchoperationen eine Lösung zum Aufrufen von AWS Lambda Funktionen für vorhandene Eingabevideos in einem S3-Quell-Bucket. Dann rufen die Lambda-Funktionen AWS Elemental MediaConvert auf, um umfangreiche Videotranskodierungsaufgaben auszuführen. Die konvertierten Ausgabe-Mediendateien werden in einem S3-Ziel-Bucket gespeichert.



Ziel

In diesem Lernprogramm erfahren Sie, wie Sie S3-Batch-Vorgänge einrichten, um eine Lambda-Funktion zum Batch-Transcodieren von Videos aufzurufen, die in einem S3-Quell-Bucket gespeichert sind. Die Lambda-Funktion ruft MediaConvert auf, um die Videos zu transcodieren. Die Ausgänge für jedes Video im S3-Quell-Bucket sind wie folgt:

- Ein HTTP Live Streaming (HLS) adaptiver Bitrate-Stream für die Wiedergabe auf Geräten mit mehreren Größen und unterschiedlichen Bandbreiten
- Eine MP4-Videodatei
- In Intervallen gesammelte Miniaturbilder

Themen

Voraussetzungen

Bevor Sie mit diesem Lernprogramm beginnen können, benötigen Sie einen Amazon S3-Quell-Bucket (z. B. **`tutorial-bucket-1`**) mit Videos, die bereits darin transcodiert werden sollen.

Sie können dem Bucket einen anderen Namen geben, wenn Sie möchten. Weitere Informationen zu Amazon S3 Bucket-Namen finden Sie unter Regeln für die Benennung von Buckets.

Lassen Sie für den S3-Quell-Bucket die Einstellungen für öffentlichen Zugriff für diesen Bucket blockieren auf die Standardeinstellungen festlegen (Alle öffentlichen Zugriffe blockieren ist aktiviert). Weitere Informationen finden Sie unter Erstellen eines Buckets.

Weitere Informationen über das Hochladen von Videos in den S3-Quell-Bucket finden Sie unter Objekte hochladen. Wenn Sie ein Video in S3 hochladen, können Sie auch Amazon S3 Transfer Acceleration nutzen, um schnelle und sichere Dateiübertragungen zu konfigurieren. Transfer Acceleration kann das Hochladen von Videos zu Ihrem S3-Bucket zur Übertragung größerer Videos über große Entfernungen beschleunigen. Weitere Informationen finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.

Schritt 1: Erstellen Sie einen S3-Bucket für Ausgabe-Mediendateien

In diesem Schritt erstellen Sie einen S3-Ziel-Bucket, um die konvertierten Ausgabe-Mediendateien zu speichern. Außerdem erstellen Sie eine CORS-Konfiguration (Cross Origin Resource Sharing), um den ursprungsübergreifenden Zugriff auf die transcodierten Mediendateien zu ermöglichen, die in Ihrem S3-Ziel-Bucket gespeichert sind.

Teilschritte

- Erstellen Sie einen Bucket für die Ausgabe-Mediendateien
- So fügen Sie einem S3-Ausgabe-Bucket eine CORS-Konfiguration hinzu:

Erstellen Sie einen Bucket für die Ausgabe-Mediendateien

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.
4. Geben Sie im Feld Bucket Name einen Namen für Ihren Bucket ein (z. B. **`tutorial-bucket-2`**).
5. Wählen Sie für Region die aus, AWS-Region in der sich der Bucket befinden soll.
6. Um den öffentlichen Zugriff auf Ihre Ausgabemediendateien zu gewährleisten, deaktivieren Sie unter Einstellungen für den öffentlichen Zugriff für diesen Bucket blockieren Alle öffentlichen Zugriffe blockieren.

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn

Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.
Wenn Sie die Einstellungen zum Blockieren des öffentlichen Zugriffs nicht löschen möchten, können Sie Amazon verwenden, CloudFront um die transcodierten Mediendateien an Viewer (Endbenutzer) zu übermitteln. Weitere Informationen finden Sie unter Tutorial: Hosten von On-Demand-Streaming-Videos mit Amazon S3 CloudFront, Amazon und Amazon Route 53.

7. Aktivieren Sie das Kontrollkästchen neben I acknowledge that the current settings may result in this bucket and the objects within becoming public (Ich bestätige, dass die aktuellen Einstellungen dazu führen können, dass dieser Bucket und die darin enthaltenen Objekte öffentlich werden).
8. Lassen Sie die restlichen Einstellungen bei den Standardwerten.
9. Wählen Sie Bucket erstellen aus.

So fügen Sie einem S3-Ausgabe-Bucket eine CORS-Konfiguration hinzu:

Die JSON-CORS-Konfiguration definiert für Client-Webanwendungen (Videoplayer in diesem Kontext), die in einer Domain geladen sind, eine Möglichkeit zur Wiedergabe transcodierter Ausgabemediendateien in einer anderen Domain.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste der Buckets den Namen des Buckets, den Sie zuvor erstellt haben (z. B. **`tutorial-bucket-2`**).
4. Wählen Sie die Registerkarte Berechtigungen.
5. Wählen Sie im Abschnitt Cross-Origin Resource Sharing (CORS) die Option Edit (Bearbeiten) aus.
6. Kopieren Sie im Textfeld CORS-Konfiguration die folgende CORS-Konfiguration und fügen Sie sie ein.

   Die CORS-Konfiguration muss im JSON-Format vorliegen. In diesem Beispiel verwendet das `AllowedOrigins`-Attribut das Platzhalterzeichen (`*`) um alle Ursprünge anzugeben. Wenn Sie Ihren spezifischen Ursprung kennen, können Sie das `AllowedOrigins`-Attribut auf Ihre

spezifische Player-URL beschränken. Weitere Informationen zum Konfigurieren von diesem und anderen Attributen finden Sie unter CORS-Konfiguration.

```
[
    {
        "AllowedOrigins": [
            "*"
        ],
        "AllowedMethods": [
            "GET"
        ],
        "AllowedHeaders": [
            "*"
        ],
        "ExposeHeaders": []

    }
]
```

7. Wählen Sie Änderungen speichern aus.

Schritt 2: Erstellen einer IAM-Rolle für MediaConvert

Um AWS Elemental MediaConvert zum Transcodieren von Eingabevideos zu verwenden, die in Ihrem S3-Bucket gespeichert sind, benötigen Sie eine AWS Identity and Access Management (IAM)-Servicerolle, um MediaConvert Berechtigungen zum Lesen und Schreiben von Videodateien aus und in Ihre S3-Quell- und Ziel-Buckets zu erteilen. Wenn Sie Transcodierungsaufträge ausführen, verwendet die MediaConvert Konsole diese Rolle.

So erstellen Sie eine IAM-Rolle für MediaConvert

1. Erstellen Sie eine IAM-Rolle mit einem von Ihnen ausgewählten Rollennamen (z. B. **`tutorial-mediaconvert-role`**). Um diese Rolle zu erstellen, folgen Sie den Schritten unter Erstellen Ihrer MediaConvert Rolle in IAM (Konsole) im AWS Elemental MediaConvert - Benutzerhandbuch.
2. Nachdem Sie die IAM-Rolle für erstellt haben MediaConvert, wählen Sie in der Liste der Rollen den Namen der Rolle aus, für MediaConvert die Sie erstellt haben (z. B. **`tutorial-mediaconvert-role`**).

3. Kopieren Sie auf der Seite Übersicht die ARN der Rolle (die mit `arn:aws:iam::` beginnt) und speichern Sie den ARN für die spätere Verwendung.

   Weitere Informationen zur Verwendung von ARNs finden Sie unter Amazon-Ressourcennamen (ARNs) in der Allgemeinen AWS -Referenz.

Schritt 3: Erstellen Sie eine IAM-Rolle für Ihre Lambda-Funktion

Um Videos mit MediaConvert und S3-Batchoperationen im Stapel zu transcodieren, verwenden Sie eine Lambda-Funktion, um diese beiden Services zum Konvertieren von Videos zu verbinden. Diese Lambda-Funktion muss über eine IAM-Rolle verfügen, die der Lambda-Funktion Berechtigungen für den Zugriff auf MediaConvert und S3-Batchoperationen gewährt.

Teilschritte

- Erstellen Sie eine IAM-Rolle für Ihre Lambda-Funktion
- Einbetten einer Inline-Richtlinie für die IAM-Rolle Ihrer Lambda -Funktion

Erstellen Sie eine IAM-Rolle für Ihre Lambda-Funktion

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles (Rollen) und dann Create Role (Rolle erstellen) aus.
3. Wählen Sie den AWS -Service-Rollentyp und wählen Sie dann unter Häufige Anwendungsfälle Lambda aus.
4. Wählen Sie Weiter: Berechtigungen aus.
5. Geben Sie auf der Seite Attach permissions (Berechtigungen anfügen) **`AWSLambdaBasicExecutionRole`** in das Suchfeld ein, um die Liste der Richtlinien zu filtern. Um die verwaltete Richtlinie AWSLambdaBasicExecutionRole an diese Rolle anzufügen, um Amazon CloudWatch Logs Schreibberechtigungen zu erteilen, aktivieren Sie das Kontrollkästchen neben AWSLambdaBasicExecutionRole.
6. Wählen Sie Weiter: Markierungen.
7. (Optional) Fügen Sie der verwalteten Richtlinie Tags hinzu.
8. Wählen Sie Weiter: Prüfen aus.

9. Geben Sie für Role name (Rollenname) den Namen **`tutorial-lambda-transcode-role`** ein.
10. Wählen Sie Create role (Rolle erstellen) aus.

Einbetten einer Inline-Richtlinie für die IAM-Rolle Ihrer Lambda -Funktion

Um Berechtigungen für die MediaConvert Ressource zu erteilen, die für die Ausführung der Lambda-Funktion erforderlich ist, müssen Sie eine Inline-Richtlinie verwenden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Wählen Sie in der Liste der Rollen den Namen der IAM-Rolle aus, die Sie vorher für Ihre Lambda-Funktion erstellt haben (z. B. **`tutorial-lambda-transcode-role`**).
4. Wählen Sie den Tab Permissions (Berechtigungen).
5. Wählen Sie Add inline Policy (Inline-Richtlinie auswählen).
6. Wählen Sie die Registerkarte JSON aus, kopieren und fügen Sie dann die folgende JSON-Richtlinie ein:

   Ersetzen Sie in der JSON-Richtlinie den Beispiel-ARN-Wert von durch `Resource` den Rollen-ARN der IAM-Rolle für , MediaConvert die Sie in Schritt 2 (z. B. **`tutorial-mediaconvert-role`**) erstellt haben.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "Logging"
        },
        {
            "Action": [
```

```
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/tutorial-mediaconvert-role"
            ],
            "Effect": "Allow",
            "Sid": "PassRole"
        },
        {
            "Action": [
                "mediaconvert:*"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "MediaConvertService"
        },
        {
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "S3Service"
        }
    ]
}
```

7. Wählen Sie Review policy (Richtlinie überprüfen) aus.
8. Geben Sie unter Name **`tutorial-lambda-policy`** ein.
9. Wählen Sie Richtlinie erstellen aus.

   Nachdem Sie eine Inline-Richtlinie erstellt haben, wird sie automatisch in Ihre IAM-Rolle Ihrer Lambda-Funktion eingebettet.

Schritt 4: Erstellen einer Lambda-Funktion für die Videotranscodierung

In diesem Abschnitt des Tutorials erstellen Sie eine Lambda-Funktion mit dem -SDK für Python zur Integration in S3-Batchoperationen und MediaConvert. Um mit der Transcodierung der

Videos zu beginnen, die bereits in Ihrem S3-Quell-Bucket gespeichert sind, führen Sie einen S3-Batchvorgang-Auftrag aus, der direkt die Lambda-Funktion für jedes Video im S3-Quell-Bucket aufruft. Anschließend sendet die Lambda-Funktion einen Transcodierungsauftrag für jedes Video an MediaConvert.

Teilschritte

- Schreiben von Lambda-Funktionscode und Erstellen eines Bereitstellungspakets
- Erstellen einer Lambda-Funktion mit einer Ausführungsrolle (Konsole)
- Stellen Sie Ihre Lambda-Funktion mit ZIP-Dateiarchiven bereit und konfigurieren Sie die Lambda-Funktion (Konsole)

Schreiben von Lambda-Funktionscode und Erstellen eines Bereitstellungspakets

1. Erstellen Sie auf Ihrem lokalen Computer einen Ordner mit dem Namen `batch-transcode`.
2. Erstellen Sie im `batch-transcode`-Ordner eine Datei mit JSON-Auftragseinstellungen. Sie können z. B. die Einstellungen in diesem Abschnitt verwenden und die Datei `job.json` benennen.

   Eine `job.json`-Datei gibt Folgendes an:

   - Welche Dateien transcodiert werden sollen
   - Wie Sie Ihre Eingabevideos transcodieren möchten
   - Welche Ausgabemediendateien Sie erstellen möchten
   - Wie die transcodierten Dateien benannt werden sollen
   - Wo die transcodierten Dateien gespeichert werden sollen
   - Welche fortschrittlichen Funktionen angewendet werden sollen und so weiter

   In diesem Tutorial wird die folgende `job.json`-Datei verwendet, um die folgenden Ausgaben für jedes Video im S3-Quell-Bucket zu erstellen:

   - Ein HTTP Live Streaming (HLS) adaptiver Bitrate-Stream für die Wiedergabe auf mehreren Geräten mit verschiedenen Größen und unterschiedlichen Bandbreiten
   - Eine MP4-Videodatei
   - In Intervallen gesammelte Miniaturbilder

Diese Beispiel-Datei `job.json` verwendet Quality-Defined Variable Bitrate (QVBR), um die Videoqualität zu optimieren. Die HLS-Ausgabe ist Apple-kompatibel (Audio ungemischt vom Video, Segmentdauer von 6 Sekunden und optimierte Videoqualität durch automatisches QVBR).

Wenn Sie die hier bereitgestellten Beispieleinstellungen nicht verwenden möchten, können Sie eine `job.json`-Spezifikation basierend auf Ihrem Anwendungsfall erstellen. Stellen Sie sicher, dass die Eingabedateien ähnliche Video- und Audiokonfigurationen aufweisen, um die Konsistenz Ihrer Ausgänge zu gewährleisten. Für beliebige Eingabedateien mit unterschiedlichen Video- und Audiokonfigurationen erstellen Sie getrennte Automatisierungen (eindeutige `job.json`-Einstellungen). Weitere Informationen finden Sie im AWS Elemental MediaConvert - Benutzerhandbuch unter Beispiel für AWS Elemental MediaConvert -Jobeinstellungen in JSON.

```
{
  "OutputGroups": [
    {
      "CustomName": "HLS",
      "Name": "Apple HLS",
      "Outputs": [
        {
          "ContainerSettings": {
            "Container": "M3U8",
            "M3u8Settings": {
              "AudioFramesPerPes": 4,
              "PcrControl": "PCR_EVERY_PES_PACKET",
              "PmtPid": 480,
              "PrivateMetadataPid": 503,
              "ProgramNumber": 1,
              "PatInterval": 0,
              "PmtInterval": 0,
              "TimedMetadata": "NONE",
              "VideoPid": 481,
              "AudioPids": [
                482,
                483,
                484,
                485,
                486,
                487,
                488,
```

```
                489,
                490,
                491,
                492
              ]
            }
          },
          "VideoDescription": {
            "Width": 640,
            "ScalingBehavior": "DEFAULT",
            "Height": 360,
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
            "Sharpness": 50,
            "CodecSettings": {
              "Codec": "H_264",
              "H264Settings": {
                "InterlaceMode": "PROGRESSIVE",
                "NumberReferenceFrames": 3,
                "Syntax": "DEFAULT",
                "Softness": 0,
                "GopClosedCadence": 1,
                "GopSize": 2,
                "Slices": 1,
                "GopBReference": "DISABLED",
                "MaxBitrate": 1200000,
                "SlowPal": "DISABLED",
                "SpatialAdaptiveQuantization": "ENABLED",
                "TemporalAdaptiveQuantization": "ENABLED",
                "FlickerAdaptiveQuantization": "DISABLED",
                "EntropyEncoding": "CABAC",
                "FramerateControl": "INITIALIZE_FROM_SOURCE",
                "RateControlMode": "QVBR",
                "CodecProfile": "MAIN",
                "Telecine": "NONE",
                "MinIInterval": 0,
                "AdaptiveQuantization": "HIGH",
                "CodecLevel": "AUTO",
                "FieldEncoding": "PAFF",
                "SceneChangeDetect": "TRANSITION_DETECTION",
                "QualityTuningLevel": "SINGLE_PASS_HQ",
                "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                "UnregisteredSeiTimecode": "DISABLED",
                "GopSizeUnits": "SECONDS",
```

```
                "ParControl": "INITIALIZE_FROM_SOURCE",
                "NumberBFramesBetweenReferenceFrames": 2,
                "RepeatPps": "DISABLED"
              }
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          },
          "OutputSettings": {
            "HlsSettings": {
              "AudioGroupId": "program_audio",
              "AudioRenditionSets": "program_audio",
              "SegmentModifier": "$dt$",
              "IFrameOnlyManifest": "EXCLUDE"
            }
          },
          "NameModifier": "_360"
        },
        {
          "ContainerSettings": {
            "Container": "M3U8",
            "M3u8Settings": {
              "AudioFramesPerPes": 4,
              "PcrControl": "PCR_EVERY_PES_PACKET",
              "PmtPid": 480,
              "PrivateMetadataPid": 503,
              "ProgramNumber": 1,
              "PatInterval": 0,
              "PmtInterval": 0,
              "TimedMetadata": "NONE",
              "TimedMetadataPid": 502,
              "VideoPid": 481,
              "AudioPids": [
                482,
                483,
                484,
                485,
                486,
                487,
                488,
                489,
                490,
```

```
                491,
                492
              ]
            }
          },
          "VideoDescription": {
            "Width": 960,
            "ScalingBehavior": "DEFAULT",
            "Height": 540,
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
            "Sharpness": 50,
            "CodecSettings": {
              "Codec": "H_264",
              "H264Settings": {
                "InterlaceMode": "PROGRESSIVE",
                "NumberReferenceFrames": 3,
                "Syntax": "DEFAULT",
                "Softness": 0,
                "GopClosedCadence": 1,
                "GopSize": 2,
                "Slices": 1,
                "GopBReference": "DISABLED",
                "MaxBitrate": 3500000,
                "SlowPal": "DISABLED",
                "SpatialAdaptiveQuantization": "ENABLED",
                "TemporalAdaptiveQuantization": "ENABLED",
                "FlickerAdaptiveQuantization": "DISABLED",
                "EntropyEncoding": "CABAC",
                "FramerateControl": "INITIALIZE_FROM_SOURCE",
                "RateControlMode": "QVBR",
                "CodecProfile": "MAIN",
                "Telecine": "NONE",
                "MinIInterval": 0,
                "AdaptiveQuantization": "HIGH",
                "CodecLevel": "AUTO",
                "FieldEncoding": "PAFF",
                "SceneChangeDetect": "TRANSITION_DETECTION",
                "QualityTuningLevel": "SINGLE_PASS_HQ",
                "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                "UnregisteredSeiTimecode": "DISABLED",
                "GopSizeUnits": "SECONDS",
                "ParControl": "INITIALIZE_FROM_SOURCE",
                "NumberBFramesBetweenReferenceFrames": 2,
```

```
                "RepeatPps": "DISABLED"
              }
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          },
          "OutputSettings": {
            "HlsSettings": {
              "AudioGroupId": "program_audio",
              "AudioRenditionSets": "program_audio",
              "SegmentModifier": "$dt$",
              "IFrameOnlyManifest": "EXCLUDE"
            }
          },
          "NameModifier": "_540"
        },
        {
          "ContainerSettings": {
            "Container": "M3U8",
            "M3u8Settings": {
              "AudioFramesPerPes": 4,
              "PcrControl": "PCR_EVERY_PES_PACKET",
              "PmtPid": 480,
              "PrivateMetadataPid": 503,
              "ProgramNumber": 1,
              "PatInterval": 0,
              "PmtInterval": 0,
              "TimedMetadata": "NONE",
              "VideoPid": 481,
              "AudioPids": [
                482,
                483,
                484,
                485,
                486,
                487,
                488,
                489,
                490,
                491,
                492
              ]
```

```
            }
          },
          "VideoDescription": {
            "Width": 1280,
            "ScalingBehavior": "DEFAULT",
            "Height": 720,
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
            "Sharpness": 50,
            "CodecSettings": {
              "Codec": "H_264",
              "H264Settings": {
                "InterlaceMode": "PROGRESSIVE",
                "NumberReferenceFrames": 3,
                "Syntax": "DEFAULT",
                "Softness": 0,
                "GopClosedCadence": 1,
                "GopSize": 2,
                "Slices": 1,
                "GopBReference": "DISABLED",
                "MaxBitrate": 5000000,
                "SlowPal": "DISABLED",
                "SpatialAdaptiveQuantization": "ENABLED",
                "TemporalAdaptiveQuantization": "ENABLED",
                "FlickerAdaptiveQuantization": "DISABLED",
                "EntropyEncoding": "CABAC",
                "FramerateControl": "INITIALIZE_FROM_SOURCE",
                "RateControlMode": "QVBR",
                "CodecProfile": "MAIN",
                "Telecine": "NONE",
                "MinIInterval": 0,
                "AdaptiveQuantization": "HIGH",
                "CodecLevel": "AUTO",
                "FieldEncoding": "PAFF",
                "SceneChangeDetect": "TRANSITION_DETECTION",
                "QualityTuningLevel": "SINGLE_PASS_HQ",
                "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                "UnregisteredSeiTimecode": "DISABLED",
                "GopSizeUnits": "SECONDS",
                "ParControl": "INITIALIZE_FROM_SOURCE",
                "NumberBFramesBetweenReferenceFrames": 2,
                "RepeatPps": "DISABLED"
              }
            },
```

```
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          },
          "OutputSettings": {
            "HlsSettings": {
              "AudioGroupId": "program_audio",
              "AudioRenditionSets": "program_audio",
              "SegmentModifier": "$dt$",
              "IFrameOnlyManifest": "EXCLUDE"
            }
          },
          "NameModifier": "_720"
        },
        {
          "ContainerSettings": {
            "Container": "M3U8",
            "M3u8Settings": {}
          },
          "AudioDescriptions": [
            {
              "AudioSourceName": "Audio Selector 1",
              "CodecSettings": {
                "Codec": "AAC",
                "AacSettings": {
                  "Bitrate": 96000,
                  "CodingMode": "CODING_MODE_2_0",
                  "SampleRate": 48000
                }
              }
            }
          ],
          "OutputSettings": {
            "HlsSettings": {
              "AudioGroupId": "program_audio",
              "AudioTrackType": "ALTERNATE_AUDIO_AUTO_SELECT_DEFAULT"
            }
          },
          "NameModifier": "_audio"
        }
      ],
      "OutputGroupSettings": {
        "Type": "HLS_GROUP_SETTINGS",
```

```
        "HlsGroupSettings": {
          "ManifestDurationFormat": "INTEGER",
          "SegmentLength": 6,
          "TimedMetadataId3Period": 10,
          "CaptionLanguageSetting": "OMIT",
          "Destination": "s3://EXAMPLE-BUCKET/HLS/",
          "DestinationSettings": {
            "S3Settings": {
              "AccessControl": {
                "CannedAcl": "PUBLIC_READ"
              }
            }
          },
          "TimedMetadataId3Frame": "PRIV",
          "CodecSpecification": "RFC_4281",
          "OutputSelection": "MANIFESTS_AND_SEGMENTS",
          "ProgramDateTimePeriod": 600,
          "MinSegmentLength": 0,
          "DirectoryStructure": "SINGLE_DIRECTORY",
          "ProgramDateTime": "EXCLUDE",
          "SegmentControl": "SEGMENTED_FILES",
          "ManifestCompression": "NONE",
          "ClientCache": "ENABLED",
          "StreamInfResolution": "INCLUDE"
        }
      }
    },
    {
      "CustomName": "MP4",
      "Name": "File Group",
      "Outputs": [
        {
          "ContainerSettings": {
            "Container": "MP4",
            "Mp4Settings": {
              "CslgAtom": "INCLUDE",
              "FreeSpaceBox": "EXCLUDE",
              "MoovPlacement": "PROGRESSIVE_DOWNLOAD"
            }
          },
          "VideoDescription": {
            "Width": 1280,
            "ScalingBehavior": "DEFAULT",
            "Height": 720,
```

```
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
            "Sharpness": 100,
            "CodecSettings": {
              "Codec": "H_264",
              "H264Settings": {
                "InterlaceMode": "PROGRESSIVE",
                "ParNumerator": 1,
                "NumberReferenceFrames": 3,
                "Syntax": "DEFAULT",
                "Softness": 0,
                "GopClosedCadence": 1,
                "HrdBufferInitialFillPercentage": 90,
                "GopSize": 2,
                "Slices": 2,
                "GopBReference": "ENABLED",
                "HrdBufferSize": 10000000,
                "MaxBitrate": 5000000,
                "ParDenominator": 1,
                "EntropyEncoding": "CABAC",
                "RateControlMode": "QVBR",
                "CodecProfile": "HIGH",
                "MinIInterval": 0,
                "AdaptiveQuantization": "AUTO",
                "CodecLevel": "AUTO",
                "FieldEncoding": "PAFF",
                "SceneChangeDetect": "ENABLED",
                "QualityTuningLevel": "SINGLE_PASS_HQ",
                "UnregisteredSeiTimecode": "DISABLED",
                "GopSizeUnits": "SECONDS",
                "ParControl": "SPECIFIED",
                "NumberBFramesBetweenReferenceFrames": 3,
                "RepeatPps": "DISABLED",
                "DynamicSubGop": "ADAPTIVE"
              }
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          },
          "AudioDescriptions": [
            {
              "AudioTypeControl": "FOLLOW_INPUT",
```

```
            "AudioSourceName": "Audio Selector 1",
            "CodecSettings": {
              "Codec": "AAC",
              "AacSettings": {
                "AudioDescriptionBroadcasterMix": "NORMAL",
                "Bitrate": 160000,
                "RateControlMode": "CBR",
                "CodecProfile": "LC",
                "CodingMode": "CODING_MODE_2_0",
                "RawFormat": "NONE",
                "SampleRate": 48000,
                "Specification": "MPEG4"
              }
            },
            "LanguageCodeControl": "FOLLOW_INPUT",
            "AudioType": 0
          }
        ]
      }
    ],
    "OutputGroupSettings": {
      "Type": "FILE_GROUP_SETTINGS",
      "FileGroupSettings": {
        "Destination": "s3://EXAMPLE-BUCKET/MP4/",
        "DestinationSettings": {
          "S3Settings": {
            "AccessControl": {
              "CannedAcl": "PUBLIC_READ"
            }
          }
        }
      }
    }
  },
  {
    "CustomName": "Thumbnails",
    "Name": "File Group",
    "Outputs": [
      {
        "ContainerSettings": {
          "Container": "RAW"
        },
        "VideoDescription": {
          "Width": 1280,
```

```
            "ScalingBehavior": "DEFAULT",
            "Height": 720,
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
            "Sharpness": 50,
            "CodecSettings": {
              "Codec": "FRAME_CAPTURE",
              "FrameCaptureSettings": {
                "FramerateNumerator": 1,
                "FramerateDenominator": 5,
                "MaxCaptures": 500,
                "Quality": 80
              }
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          }
        }
      ],
      "OutputGroupSettings": {
        "Type": "FILE_GROUP_SETTINGS",
        "FileGroupSettings": {
          "Destination": "s3://EXAMPLE-BUCKET/Thumbnails/",
          "DestinationSettings": {
            "S3Settings": {
              "AccessControl": {
                "CannedAcl": "PUBLIC_READ"
              }
            }
          }
        }
      }
    }
  ],
  "AdAvailOffset": 0,
  "Inputs": [
    {
      "AudioSelectors": {
        "Audio Selector 1": {
          "Offset": 0,
          "DefaultSelection": "DEFAULT",
          "ProgramSelection": 1
```

```
        }
      },
      "VideoSelector": {
        "ColorSpace": "FOLLOW"
      },
      "FilterEnable": "AUTO",
      "PsiControl": "USE_PSI",
      "FilterStrength": 0,
      "DeblockFilter": "DISABLED",
      "DenoiseFilter": "DISABLED",
      "TimecodeSource": "EMBEDDED",
      "FileInput": "s3://EXAMPLE-INPUT-BUCKET/input.mp4"
    }
  ]
}
```

3. Erstellen Sie im `batch-transcode`-Ordner eine Datei mit einer Lambda-Funktion. Sie können das folgende Python-Beispiel verwenden und die Datei `convert.py` nennen.

   S3-Batchvorgänge sendet bestimmte Aufgabendaten an eine Lambda-Funktion und erfordert Ergebnisdaten zurück. Anforderungs- und Antwortbeispielen für die Lambda-Funktion, Informationen über Antwort- und Ergebniscodes sowie Beispiel-Lambda-Funktionen für S3-Batchvorgänge finden Sie unter Aufrufen einer AWS Lambda-Funktion.

```
import json
import os
from urllib.parse import urlparse
import uuid
import boto3

"""
When you run an S3 Batch Operations job, your job
invokes this Lambda function. Specifically, the Lambda function is
invoked on each video object listed in the manifest that you specify
for the S3 Batch Operations job in Step 5.

Input parameter "event": The S3 Batch Operations event as a request
                         for the Lambda function.

Input parameter "context": Context about the event.

Output: A result structure that Amazon S3 uses to interpret the result
```

```
        of the operation. It is a job response returned back to S3 Batch
 Operations.
"""
def handler(event, context):

    invocation_schema_version = event['invocationSchemaVersion']
    invocation_id = event['invocationId']
    task_id = event['tasks'][0]['taskId']

    source_s3_key = event['tasks'][0]['s3Key']
    source_s3_bucket = event['tasks'][0]['s3BucketArn'].split(':::')[-1]
    source_s3 = 's3://' + source_s3_bucket + '/' + source_s3_key

    result_list = []
    result_code = 'Succeeded'
    result_string = 'The input video object was converted successfully.'

    # The type of output group determines which media players can play
    # the files transcoded by MediaConvert.
    # For more information, see Creating outputs with AWS Elemental MediaConvert.
    output_group_type_dict = {
        'HLS_GROUP_SETTINGS': 'HlsGroupSettings',
        'FILE_GROUP_SETTINGS': 'FileGroupSettings',
        'CMAF_GROUP_SETTINGS': 'CmafGroupSettings',
        'DASH_ISO_GROUP_SETTINGS': 'DashIsoGroupSettings',
        'MS_SMOOTH_GROUP_SETTINGS': 'MsSmoothGroupSettings'
    }

    try:
        job_name = 'Default'
        with open('job.json') as file:
            job_settings = json.load(file)

        job_settings['Inputs'][0]['FileInput'] = source_s3

        # The path of each output video is constructed based on the values of
        # the attributes in each object of OutputGroups in the job.json file.
        destination_s3 = 's3://{0}/{1}/{2}' \
            .format(os.environ['DestinationBucket'],
                    os.path.splitext(os.path.basename(source_s3_key))[0],
                    os.path.splitext(os.path.basename(job_name))[0])

        for output_group in job_settings['OutputGroups']:
            output_group_type = output_group['OutputGroupSettings']['Type']
```

```
            if output_group_type in output_group_type_dict.keys():
                output_group_type = output_group_type_dict[output_group_type]
                output_group['OutputGroupSettings'][output_group_type]
['Destination'] = \
                    "{0}{1}".format(destination_s3,
                                    urlparse(output_group['OutputGroupSettings']
[output_group_type]['Destination']).path)
            else:
                raise ValueError("Exception: Unknown Output Group Type {}."
                                 .format(output_group_type))

        job_metadata_dict = {
            'assetID': str(uuid.uuid4()),
            'application': os.environ['Application'],
            'input': source_s3,
            'settings': job_name
        }

        region = os.environ['AWS_DEFAULT_REGION']
        endpoints = boto3.client('mediaconvert', region_name=region) \
            .describe_endpoints()
        client = boto3.client('mediaconvert', region_name=region,
                              endpoint_url=endpoints['Endpoints'][0]['Url'],
                              verify=False)

        try:
            client.create_job(Role=os.environ['MediaConvertRole'],
                              UserMetadata=job_metadata_dict,
                              Settings=job_settings)
        # You can customize error handling based on different error codes that
        # MediaConvert can return.
        # For more information, see MediaConvert error codes.
        # When the result_code is TemporaryFailure, S3 Batch Operations retries
        # the task before the job is completed. If this is the final retry,
        # the error message is included in the final report.
        except Exception as error:
            result_code = 'TemporaryFailure'
            raise

    except Exception as error:
        if result_code != 'TemporaryFailure':
            result_code = 'PermanentFailure'
        result_string = str(error)
```

```
    finally:
        result_list.append({
            'taskId': task_id,
            'resultCode': result_code,
            'resultString': result_string,
        })

    return {
        'invocationSchemaVersion': invocation_schema_version,
        'treatMissingKeyAs': 'PermanentFailure',
        'invocationId': invocation_id,
        'results': result_list
    }
```

4. Um ein Bereitstellungspaket mit `convert.py` und `job.json` als `.zip`-Datei mit dem Namen `lambda.zip` zu erstellen, öffnen Sie in Ihrem lokalen Terminal den `batch-transcode`-Ordner, den Sie bereits erstellt haben, und führen Sie den folgenden Befehl aus.

   Führen Sie für macOS-Benutzer den folgenden Befehl aus:

   ```
   zip -r lambda.zip convert.py job.json
   ```

   Führen Sie für Windows-Benutzer die folgenden Befehle aus:

   ```
   powershell Compress-Archive convert.py lambda.zip
   ```

   ```
   powershell Compress-Archive -update job.json lambda.zip
   ```

Erstellen einer Lambda-Funktion mit einer Ausführungsrolle (Konsole)

1. Öffnen Sie die - AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie im linken Navigationsbereich die Option Functions (Funktionen) aus.
3. Wählen Sie Create function (Funktion erstellen).
4. Wählen Sie Author from scratch aus.
5. Führen Sie unter Basic information (Grundlegende Informationen) die folgenden Schritte aus:

   a. Geben Sie für Function name (Funktionsname) **`tutorial-lambda-convert`** ein.

   b. Wählen Sie für Laufzeit Python 3.8 oder eine höhere Version von Python aus.
6. Wählen Sie Ändern der standardmäßigen Ausführungsrolle und wählen Sie unter Ausführungsrolle Verwenden einer vorhandenen Rolle.
7. Wählen Sie unter Bestehende Rolle den Namen der IAM-Rolle aus, die Sie für Ihre Lambda-Funktion in Schritt 3 (zum Beispiel **`tutorial-lambda-transcode-role`**) erstellt haben.
8. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt.
9. Wählen Sie Funktion erstellen aus.

Stellen Sie Ihre Lambda-Funktion mit ZIP-Dateiarchiven bereit und konfigurieren Sie die Lambda-Funktion (Konsole)

1. Wählen Sie im Abschnitt Code-Quelle der Seite für die Lambda-Funktion, die Sie erstellt haben (z. B. **`tutorial-lambda-convert`**), Hochladen von und dann ZIP-Datei aus.
2. Wählen Sie Upload (Hochladen) aus, um Ihre lokale `.zip`-Datei auszuwählen.
3. Wählen Sie die zuvor erstellte `lambda.zip`-Datei und klicken Sie auf Öffnen.
4. Wählen Sie Speichern aus.
5. Wählen Sie im Abschnitt Runtime settings (Laufzeiteinstellungen) die OptionBearbeiten aus.
6. Um der Lambda-Laufzeit mitzuteilen, welche Handler-Methode in Ihrem Lambda-Funktionscode aufgerufen werden soll, geben Sie **`convert.handler`** in das Feld Handler ein.

   Wenn Sie eine Funktion in Python konfigurieren, besteht der Wert der Handler-Einstellung aus dem Dateinamen und dem Namen des Handler-Moduls, getrennt durch einen Punkt (`.`). Beispielsweise ruft `convert.handler` die Methode `handler` auf, die in der Datei `convert.py` definiert ist.
7. Wählen Sie Speichern.
8. Wählen Sie auf der Seite der Lambda-Funktion den Tab Konfiguration. Im linken Navigationsbereich auf der Registerkarte Konfiguration wählen Sie Umgebungsvariablen und wählen dann Bearbeiten aus.
9. Wählen Sie Umgebungsvariablen hinzufügen aus. Geben Sie dann den angegebenen Schlüssel und Wert für jede der folgenden Umgebungsvariablen ein:

   - Schlüssel: **`DestinationBucket`** Wert: **`tutorial-bucket-2`**

     Dieser Wert ist der S3 Bucket für Ausgabe-Mediendateien, die Sie in Schritt 1 erstellt haben.

- Schlüssel: **`MediaConvertRole`** Wert: **`arn:aws:iam::`*`111122223333`*`:role/tutorial-mediaconvert-role`**

  Dieser Wert ist der ARN der IAM-Rolle für MediaConvert , die Sie in Schritt 2 erstellt haben. Stellen Sie sicher, dass Sie diesen ARN durch den tatsächlichen ARN Ihrer IAM-Rolle ersetzen.

- Schlüssel: **`Application`** Wert: **`Batch-Transcoding`**

  Dieser Wert ist der Name der Anwendung.

10. Wählen Sie Speichern.
11. (Optional) Wählen Sie im Tab Konfigurationim Abschnitt Allgemeine Konfiguration des linken Navigationsbereichs Bearbeiten aus. Geben Sie im Timeout-Feld **`2`** m **`0`** s ein. Wählen Sie dann Save (Speichern) aus.

    Timeout ist die Zeitspanne, die Lambda einer Funktion für einen Aufruf zulässt, bevor diese gestoppt wird. Der Standardwert ist 3 Sekunden. Die Preise basieren auf dem konfigurierten Arbeitsspeicher und der Zeit, für die der Code ausgeführt wird. Weitere Informationen finden Sie unter AWS Lambda Preise.

Schritt 5: Konfigurieren des Amazon S3-Bestands für Ihren S3-Quell-Bucket

Nachdem Sie die Transcodierungs-Lambda-Funktion eingerichtet haben, erstellen Sie einen S3-Batchvorgangsauftrag, um eine Reihe von Videos zu transcodieren. Zuerst benötigen Sie eine Liste von Eingabe-Videoobjekten, für die S3-Batch-Vorgänge die angegebene Transcodierungsaktion ausführen soll. Um eine Liste von Eingabe-Videoobjekten abzurufen, können Sie einen S3-Inventarbericht für Ihren S3-Quell-Bucket erstellen (z. B. **`tutorial-bucket-1`**).

Teilschritte

- Erstellen und Konfigurieren eines Buckets für S3-Inventarberichte für Eingabevideos
- Konfigurieren des Amazon S3-Bestands für Ihren S3-Quell-Bucket
- Überprüfen Sie den Lagerbestandsbericht für Ihren S3-Videoquellen-Bucket

Erstellen und Konfigurieren eines Buckets für S3-Inventarberichte für Eingabevideos

Um einen S3-Bestandslistenbericht zu speichern, der die Objekte des S3-Quell-Buckets auflistet, erstellen Sie einen S3-Inventarziel-Bucket und konfigurieren Sie dann eine Bucket-Richtlinie, damit der Bucket Inventardateien in den S3-Quell-Bucket schreibt.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.
4. Geben Sie im Feld Bucket Name einen Namen für Ihren Bucket ein (z. B. **`tutorial-bucket-3`**).
5. Wählen Sie für die ausAWS-Region, AWS-Region in der sich der Bucket befinden soll.

   Der Bestandsziel-Bucket muss sich in derselben befinden AWS-Region wie der Quell-Bucket, in dem Sie S3 Inventory einrichten. Der Bestands-Ziel-Bucket kann sich in einem anderem AWS-Konto befinden.
6. Behalten Sie die Einstellungen für öffentlichen Zugriff für diesen Bucket blockieren auf die Standardeinstellungen gestellt (Alle öffentlichen Zugriffe blockieren ist aktiviert).
7. Lassen Sie die restlichen Einstellungen auf die Standardwerte eingestellt.
8. Wählen Sie Bucket erstellen aus.
9. Wählen Sie in der Liste der Buckets den Namen des Buckets, den Sie gearde erstellt haben (z. B. **`tutorial-bucket-3`**).
10. Um Amazon S3 die Berechtigung zu erteilen, Daten für die Bestandsberichte in den S3-Basisziel-Bucket zu schreiben, wählen Sie das Tab Berechtigungen aus.
11. Scrollen Sie nach unten zum Abschnitt Bucket-Richtlinie und wählen Sie Bearbeiten aus. Die Seite Bucket-Richtlinie wird geöffnet.
12. Um Berechtigungen für S3-Inventar zu erteilen, fügen Sie im Feld Richtlinie die folgende Bucket-Richtlinie ein.

    Ersetzen Sie die drei Beispielwerte durch die folgenden Werte:

    - Der Name des Buckets, den Sie erstellt haben, um die Bestandsberichte zu speichern (z. B. *`tutorial-bucket-3`*).
    - Der Name des Quell-Buckets, in dem die Eingabe-Videos gespeichert werden (z. B. *`tutorial-bucket-1`*).

- Die AWS-Konto ID, die Sie zum Erstellen des S3-Videoquellen-Buckets verwendet haben (z. B. *111122223333*.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"InventoryAndAnalyticsExamplePolicy",
      "Effect":"Allow",
      "Principal": {"Service": "s3.amazonaws.com"},
      "Action":"s3:PutObject",
      "Resource":["arn:aws:s3:::tutorial-bucket-3/*"],
      "Condition": {
          "ArnLike": {
              "aws:SourceArn": "arn:aws:s3:::tutorial-bucket-1"
           },
         "StringEquals": {
             "aws:SourceAccount": "111122223333",
             "s3:x-amz-acl": "bucket-owner-full-control"
           }
       }
    }
  ]
}
```

13. Wählen Sie Änderungen speichern aus.

Konfigurieren des Amazon S3-Bestands für Ihren S3-Quell-Bucket

Sie müssen S3-Inventar für Ihren S3-Videoquellen-Bucket konfigurieren, um eine flache Dateiliste mit Videoobjekten und Metadaten zu generieren. Diese geplanten Bestandsberichte können alle Objekte im Bucket oder Objekte enthalten, die nach einem gemeinsamen Präfix gruppiert sind. In diesem Tutorial enthält der S3-Bestandsbericht alle Video-Objekte in Ihrem S3-Quell-Bucket.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.

3. Um einen S3-Inventarbericht der Eingabe-Videos in Ihrem S3-Quell-Bucket zu konfigurieren, wählen Sie in der Liste Buckets zum Beispiel den Namen des S3-Quell-Buckets aus (z. B. **`tutorial-bucket-1`**).
4. Wählen Sie den Tab Management.
5. Blättern Sie nach unten bis zur Konfigurationen für Bestandserfassung und wählen Sie Erstellen der Bestandskonfiguration aus.
6. Geben Sie für Name der Bestandserfassung z. B. einen Namen ein (z. B. **`tutorial-inventory-config`**).
7. Wählen Sie unter Bestandserfassung Nur aktuelle Version für Objekt-Versionen und lassen Sie andere Einstellungen der Bestandserfassung auf die Standardwerte für dieses Tutorial eingestellt.
8. Wählen Sie im Abschnitt Berichtsdetails für Ziel-Bucket Dieses Konto aus.
9. Wählen Sie für Ziel S3 durchsuchen und wählen Sie den Ziel-Bucket aus, den Sie bereits erstellt haben, um die Lagerbestandsberichte zu speichern (z. B .**`tutorial-bucket-3`**). Wählen Sie dann Pfad wählen aus.

   Der Bestandsziel-Bucket muss sich in derselben befinden AWS-Region wie der Quell-Bucket, in dem Sie S3 Inventory einrichten. Der Bestands-Ziel-Bucket kann sich in einem anderem AWS-Konto befinden.

   Unter dem Bucket-Feld Destination (Ziel) wird die Destination bucket permission (Ziel-Bucket-Berechtigung) der Bestands-Ziel-Bucket-Richtlinie hinzugefügt, damit Amazon S3 Daten in diesen Bestands-Ziel-Bucket platzieren kann. Weitere Informationen finden Sie unter Erstellen einer Ziel-Bucket-Richtlinie.
10. Wählen Sie für Häufigkeit Täglich aus.
11. Für das Ausgabeformat wählen Sie CSV.
12. Wählen Sie für Status die Option Aktiviert.
13. Wählen Sie im Abschnitt Serverseitige Verschlüsselung Deaktivieren für dieses Tutorial.

    Weitere Informationen finden Sie unter Konfigurieren des Bestands mit der S3-Konsole und Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres vom Kunden verwalteten Schlüssels für die Verschlüsselung.
14. Wählen Sie unter Zusätzliche Felder – optional Größe, Letzte Änderung, und Speicherklasse aus.
15. Wählen Sie Create (Erstellen) aus.

Weitere Informationen finden Sie unter Konfigurieren des Bestands mit der S3-Konsole.

Überprüfen Sie den Lagerbestandsbericht für Ihren S3-Videoquellen-Bucket

Wenn ein Bestandsbericht veröffentlicht wird, werden die Manifestdateien an den S3-Bestands-Ziel-Bucket gesendet.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Video-Quell-Buckets (z. B. **`tutorial-bucket-1`**).
4. Wählen Sie Management (Verwaltung) aus.
5. Um zu sehen, ob Ihr S3-Bestandsbericht für das Erstellen eines S3-Batchvorgänge-Auftrags in Schritt 7 bereit ist, überprüfen Sie unter Bestandskonfigurationen, ob die Schaltfläche Auftrag aus Manifest erstellen aktiviert ist.

   > **Note**
   >
   > Es kann bis zu 48 Stunden dauern, bis der erste Bestandsbericht bereitgestellt wird. Wenn die Schaltfläche Auftrag aus Manifest erstellendeaktiviert ist, wurde der erste Bestandsbericht nicht zugestellt. Warten Sie, bis der erste Bestandsbericht zugestellt wurde und die Schaltfläche Auftrag aus Manifest erstellen aktiviert ist, bevor Sie einen S3-Batchvorgänge-Auftrag in Schritt 7 erstellen.

6. Um einen S3-Bestandsbericht (`manifest.json`) in der die Spalte Ziel zu überprüfen, wählen Sie den Namen des Bestandsziels aus, den Sie bereits zum Speichern von Bestandsberichten erstellt haben (z. B. **`tutorial-bucket-3`**).
7. Wählen Sie auf der Registerkarte Objekte den vorhandenen Ordner mit dem Namen Ihres S3-Quell-Buckets (z. B. **`tutorial-bucket-1`**). Wählen Sie dann den Namen aus, den Sie in Name der Bestandserfassung eingegeben haben, als Sie die Inventarkonfiguration früher erstellt haben (z. B. **`tutorial-inventory-config`**).

   Sie können eine Liste von Ordnern mit den Generierungsdaten der Berichte als Namen sehen.
8. Um den täglichen S3-Bestandsbericht an einem bestimmten Datum zu überprüfen, wählen Sie den Ordner mit dem entsprechenden Namen des Generierungsdatums aus und wählen Sie dann `manifest.json` aus.

9. Um die Details des Bestandsberichts zu einem bestimmten Datum zu überprüfen, wählen Sie auf der Seite manifest.json Herunterladen oder Öffnen aus.

Schritt 6: Erstellen einer IAM-Rolle für S3-Batchvorgänge

Um S3-Batchvorgänge für die Batch-Transcodierung verwenden zu können, müssen Sie zunächst eine IAM-Rolle erstellen, um Amazon S3 Berechtigungen zum Ausführen von S3-Batchvorgängen zu geben.

Teilschritte

- Erstellen einer IAM-Richtlinie für S3-Batchvorgänge
- Erstellen Sie eine S3-Batchvorgänge-IAM-Rolle und fügen Sie Berechtigungsrichtlinien an

Erstellen einer IAM-Richtlinie für S3-Batchvorgänge

Sie müssen eine IAM-Richtlinie erstellen, die S3-Batchvorgängen die Berechtigung zum Lesen des Eingabemanifests, zum Aufrufen der Lambda-Funktion und zum Schreiben desAbschlussberichts des S3-Batchvorgänge-Auftrags erteilt.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich die Option Policies (Richtlinien) aus.
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie den Tab JSON.
5. Fügen Sie die folgende JSON-Richtlinie in das JSON-Textfeld ein.

   Ersetzen Sie in der JSON-Richtlinie die vier Beispielwerte durch die folgenden Werte:

   - Der Name des Quell-Buckets, in dem Ihre Eingabe-Videos gespeichert werden (z. B. *`tutorial-bucket-1`*).
   - Der Name des Bestands-Ziel-Buckets, den Sie in Schritt 5 erstellt haben, um `manifest.json`-Dateien (z. B. *`tutorial-bucket-3`*) zu speichern.
   - Der Name des Buckets, den Sie in Schritt 1 erstellt haben, um Ausgabe-Mediendateien zu speichern (z. B. *`tutorial-bucket-2`*). In diesem Tutorial legen wir Auftragsvervollständigungsberichte in den Ziel-Bucket für Ausgabe-Mediendateien.

- Die Rollen-ARN der Lambda -Funktion, die Sie im AbschnittSchritt 4 erstellt haben. Um die Rolle ARN der Lambda-Funktion zu finden und zu kopieren, gehen Sie wie folgt vor:
  - Öffnen Sie in einem neuen Browser-Tab die Seite Funktionen der Lambda-Konsole bei https://console.aws.amazon.com/lambda/home#/functions.
  - Wählen Sie in der Liste Funktionen den Namen der Lambda-Funktion aus, die Sie in Schritt 4 (zum Beispiel **`tutorial-lambda-convert`**) erstellt haben.
  - Klicken Sie auf Copy ARN (ARN kopieren).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3Get",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::tutorial-bucket-1/*",
                "arn:aws:s3:::tutorial-bucket-3/*"
            ]
        },
        {
            "Sid": "S3PutJobCompletionReport",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::tutorial-bucket-2/*"
        },
        {
            "Sid": "S3BatchOperationsInvokeLambda",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:us-west-2:111122223333:function:tutorial-lambda-
convert"
            ]
        }
```

```
    ]
}
```

6. Wählen Sie Next: Tags (Weiter: Tags) aus.
7. Klicken Sie auf Weiter: Prüfen.
8. Geben Sie im Feld Name **`tutorial-s3batch-policy`** ein.
9. Wählen Sie Richtlinie erstellen aus.

Erstellen Sie eine S3-Batchvorgänge-IAM-Rolle und fügen Sie Berechtigungsrichtlinien an

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles (Rollen) und dann Create Role (Rolle erstellen) aus.
3. Wählen Sie den AWS-Service-Rollentyp und danach den Service S3 aus.
4. Wählen Sie unter Ihren Anwendungsfall auswählen S3-Batch-Vorgänge.
5. Wählen Sie Weiter: Berechtigungen aus.
6. Geben Sie unter Anfügen von Berechtigungsrichtlinien den Namen der vorher erstellten IAM-Richtlinie ein (z. B.**`tutorial-s3batch-policy`**), um die Liste der Richtlinien zu filtern. Aktivieren Sie das Kontrollkästchen neben dem Namen der Richtlinie (z. B. **`tutorial-s3batch-policy`**).
7. Wählen Sie Next: Tags (Weiter: Tags) aus.
8. Wählen Sie Weiter: Prüfen aus.
9. Geben Sie für Role name (Rollenname) den Namen **`tutorial-s3batch-role`** ein.
10. Wählen Sie Create role (Rolle erstellen) aus.

    Nachdem Sie die IAM-Rolle für S3-Batchvorgänge erstellt haben, wird die folgende Vertrauensrichtlinie automatisch an die Rolle angefügt. Diese Vertrauensrichtlinie ermöglicht dem S3-Batchoperationen-Service-Prinzipal, die IAM-Rolle zu übernehmen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
```

```
            "Principal":{
               "Service":"batchoperations.s3.amazonaws.com"
            },
            "Action":"sts:AssumeRole"
         }
   ]
}
```

Schritt 7: Einrichten und Ausführen eines Auftrags für S3-Batchvorgänge

Um einen S3-Batchvorgänge-Auftrag zu erstellen, um die Eingabevideos in Ihrem S3-Quell-Bucket zu verarbeiten, müssen Sie Parameter für diesen bestimmten Auftrag angeben.

> **Note**
>
> Um mit der Erstellung eines S3-Batchvorgänge-Auftrags zu beginnen, stellen Sie sicher, dass die Schaltfläche Auftrag aus Manifest erstellen aktiviert ist. Weitere Informationen finden Sie unter Überprüfen Sie den Lagerbestandsbericht für Ihren S3-Videoquellen-Bucket. Wenn die Schaltfläche Auftrag aus Manifest erstellen deaktiviert ist, wurde der erste Inventarbericht nicht übermittelt und Sie müssen warten, bis die Schaltfläche aktiviert ist. Nachdem Sie Amazon S3-Bestand für Ihren S3-Quell-Bucket in Schritt 5 konfiguriert haben, kann es bis zu 48 Stunden dauern, bis der erste Bestandsbericht übermittelt wird.

Teilschritte

- Erstellen eines S3-Batchvorgangsauftrags
- Führen Sie den S3-Batchvorgänge-Auftrag aus, um Ihre Lambda-Funktion aufzurufen.
- (Optional) Prüfen Sie Ihren Abschlussbericht
- (Optional) Überwachen Sie jeden Lambda-Aufruf in der Lambda-Konsole
- (Optional) Überwachen Sie jeden MediaConvert Videotranscodierungsauftrag in der MediaConvert Konsole

Erstellen eines S3-Batchvorgangsauftrags

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Klicken Sie im linken Navigationsbereich auf Batchvorgänge.
3. Wählen Sie Job erstellen aus.
4. Wählen Sie für AWS-Region die Region aus, in der Sie Ihren Auftrag erstellen möchten.

   In diesem Tutorial müssen Sie den S3-Batchvorgänge-Auftrag zum Aufrufen einer Lambda-Funktion in derselben Region wie den S3-Videoquellen-Bucket erstellen, in dem sich die im Manifest referenzierten Objekte befinden.
5. Gehen Sie im Abschnitt Manifest wie folgt vor:

   a. Wählen Sie für Manifestformat S3-Inventory-Bericht (manifest.json) aus.

   b. Wählen Sie für Manifest-Objekt S3 durchsuchen, um den Bucket zu finden, den Sie in Schritt 5 zum Speichern von Bestandsberichten (z. B. **`tutorial-bucket-3`**) erstellt haben. Navigieren Sie auf der Seite Manifest-Objekt durch die Objektnamen, bis Sie eine `manifest.json`-Datei für ein bestimmtes Datum finden. Diese Datei listet die Informationen zu allen Videos auf, die Sie Batch-Transcodieren möchten. Wenn Sie die `manifest.json`-Datei gefunden haben, die Sie verwenden möchten, wählen Sie die Optionsschaltfläche daneben. Wählen Sie dann Pfad wählen aus.

   c. (Optional) Geben Sie für Versions-ID des Manifestobjekts – optional die Versions-ID des Manifestobjekts ein, wenn Sie eine andere Version als die aktuelle verwenden möchten.
6. Wählen Sie Next (Weiter).
7. Um die Lambda-Funktion zum Transcodieren aller Objekte zu verwenden, die in der ausgewählten `manifest.json`-Datei aufgelistet sind, wählen Sie unter Vorgangstyp AWS Lambda -Funktion aufrufen aus.
8. Führen Sie im Abschnitt Aufrufen einer Lambda-Funktion Folgendes aus:

   a. Klicken Sie auf Wählen Sie aus den Funktionen Ihres Kontos aus.

   b. Wählen Sie für Lambda-Funktion die Lambda-Funktion aus, die Sie in Schritt 4 (zum Beispiel **`tutorial-lambda-convert`**) erstellt haben.

   c. Behalten Sie für Version der Lambda-Funktion den Standardwert $LATEST bei.
9. Wählen Sie Weiter aus. Die Seite Konfigurieren zusätzlicher Optionen wird geöffnet.
10. Behalten Sie im Abschnitt Zusätzliche Optionen die Standardeinstellungen bei.

    Weitere Informationen zu diesen Optionen finden Sie unter Batch-Vorgangsauftrag-Anforderungselemente.

11. Wählen Sie im Abschnitt Abschlussbericht für Pfad zum Ziel des Abschlussberichts S3 durchsuchen aus. Finden Sie den Bucket, den Sie in Schritt 1 für die Ausgabe-Mediendateien erstellte haben (z. B. **`tutorial-bucket-2`**). Wählen Sie das Optionsfeld neben dem Namen dieses Buckets aus. Wählen Sie dann Pfad wählen aus.

    Lassen Sie die restlichen Einstellungen des Abschlussberichts auf die Standardwerte eingestellt. Weitere Informationen zu den Einstellungen des Fertigstellungsberichts finden Sie unter Batch-Vorgangsauftrag-Anforderungselemente. Ein Abschlussbericht führt eine Aufzeichnung der Details des Auftrags und der ausgeführten Vorgänge.

12. Wählen Sie im Abschnitt Berechtigungen Wählen Sie aus vorhandenen IAM-Rollen aus. Wählen Sie für IAM-Rolle die IAM-Rolle für Ihren S3-Batch-Vorgänge-Auftrag aus, den Sie im Abschnitt Schritt 6 (z. B. **`tutorial-s3batch-role`**) erstellt haben.
13. Wählen Sie Weiter aus.
14. Überprüfen Sie die Einstellungen auf der Seite Review. Wählen Sie dann Auftrag erstellen aus.

    Nachdem S3 das Manifest Ihres S3-Batchvorgänge-Auftrags gelesen hat, wechselt es den Auftrag in den Zustand Awaiting your confirmation to run (Wartet auf Ihre Bestätigung zur Ausführung). Um Aktualisierungen des Auftragsstatus anzuzeigen, aktualisieren Sie die Seite. Sie können Ihren Job erst ausführen, wenn der Status lautet Wartet auf Ihre Bestätigung zur Ausführung.

Führen Sie den S3-Batchvorgänge-Auftrag aus, um Ihre Lambda-Funktion aufzurufen.

Führen Sie Ihren Batchvorgänge-Auftrag aus, um Ihre Lambda-Funktion für die Videotranscodierung aufzurufen. Wenn Ihr Auftrag fehlschlägt, können Sie den Abschlussbericht überprüfen, um die Ursache zu ermitteln.

So führen Sie den S3-Batchvorgänge-Auftrag aus

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie im linken Navigationsbereich auf Batchvorgänge.
3. Wählen Sie aus der Liste Aufträge die Auftrags-ID des Auftrags in der ersten Zeile aus, bei der es sich um den vorher erstellten S3-Batchvorgänge-Auftrag handelt.
4. Wählen Sie Auftrag ausführen aus.

5. Überprüfen Sie die Auftrags-Parameter erneut, und bestätigen Sie, dass der Wert für Insgesamt im Manifest aufgelistete Objekte mit der Anzahl der Objekte im Manifest identisch ist. Wählen Sie dann Auftrag ausführen.

   Ihre Seite des S3-Batchvorgänge-Auftrags wird geöffnet.

6. Nachdem der Auftrag gestartet wurde, kontrollieren Sie unter Status den Fortschritt Ihres S3-Batchvorgänge-Auftrags, wie z. B. Status, % abgeschlossen, Gesamt erfolgreich (Rate), Gesamt fehlgeschlagen (Rate), Beendigungsdatum, und Grund für die Beendigung.

   Wenn der S3-Batchvorgänge-Auftrag abgeschlossen ist, zeigen Sie die Daten auf Ihrer Auftragaseite an, um zu bestätigen, dass der Auftrag wie erwartet abgeschlossen wurde.

   Wenn mehr als 50 Prozent der Objektvorgänge eines S3-Batchvorgänge-Auftrags fehlschlagen, nachdem mehr als 1000 Vorgänge versucht wurden, schlägt der Auftrag automatisch fehl. Um Ihren Abschlussbericht zu überprüfen, um die Ursache der Fehler zu ermitteln, verwenden Sie das folgenden optionale Verfahren.

(Optional) Prüfen Sie Ihren Abschlussbericht

Sie können Ihren Abschlussbericht verwenden, um festzustellen, welche Objekte fehlgeschlagen sind und welche Ursache der Fehler sind.

Um Ihren Abschlussbericht auf Details zu fehlgeschlagenen Objekten zu überprüfen:

1. Blättern Sie auf der Seite Ihres S3-Batchvorgänge-Auftrags zum Abschnitt Abschlussbericht und wählen Sie den Link unter Ziel des Abschlussberichts aus.

   Die Seite des S3-Ausgabe-Ziel-Buckets wird geöffnet.

2. Wählen Sie auf der Registerkarte Objekte auf den Ordner, dessen Name mit der Auftrags-ID des vorher erstellten S3-Batchvorgänge-Auftrags endet.
3. Klicken Sie auf Ergebnis/se.
4. Aktivieren Sie das Kontrollkästchen neben der `.csv`-Datei.
5. Wählen Sie Öffnen oder Herunterladen, um den Auftragsbericht anzuzeigen.

(Optional) Überwachen Sie jeden Lambda-Aufruf in der Lambda-Konsole

Nachdem der S3-Batchvorgänge-Auftrag gestartet wurde, ruft der Auftrag, die Lambda-Funktion für jedes Eingabe-Video-Objekt auf. S3 schreibt Protokolle jedes Lambda-Aufrufs in CloudWatch Protokolle. Sie können das Monitoring-Dashboard der Lambda-Konsole verwenden, um Ihre Lambda-Funktion zu überwachen.

1. Öffnen Sie die - AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie im linken Navigationsbereich die Option Functions (Funktionen) aus.
3. Wählen Sie in der Liste Funktionen den Namen der Lambda-Funktion aus, die Sie in Schritt 4 (zum Beispiel **`tutorial-lambda-convert`**) erstellt haben.
4. Wählen Sie den Tab Überwachung.
5. Finden Sie unter Metriken die Laufzeitmetriken für Ihre Lambda-Funktion.
6. Zeigen Sie unter Protokolle die Protokolldaten für jeden Lambda-Aufruf über CloudWatch Logs Insights an.

> **Note**
>
> Wenn Sie S3-Batch-Vorgänge mit einer Lambda-Funktion verwenden, wird die Lambda-Funktion für jedes Objekt aufgerufen. Wenn Ihr S3-Batchvorgänge-Auftrag groß ist, kann er mehrere Lambda-Funktionen gleichzeitig aufrufen, was zu einer Spitze der Lambda-Gleichzeitigkeit führt.
> Jedes AWS-Konto hat ein Lambda-Kontingent für Gleichzeitigkeit pro Region. Weitere Informationen finden Sie unter Funktionsskalierung AWS Lambda im AWS Lambda -Entwicklerhandbuch. Eine bewährte Methode für die Verwendung von Lambda-Funktionen mit S3-Batchvorgängen besteht darin, eine Gleichzeitigkeitsgrenze für die Lambda-Funktion selbst festzulegen. Ein Parallelitäts-Limit aufzustellen verhindert, dass Ihr Auftrag den größten Teil Ihrer Lambda-Parallelität verbraucht und möglicherweise andere Funktionen in Ihrem Konto drosselt. Weitere Informationen finden Sie unter Verwalten reservierter Parallelität von Lambda im AWS Lambda -Entwicklerhandbuch.

(Optional) Überwachen Sie jeden MediaConvert Videotranscodierungsauftrag in der MediaConvert Konsole

Ein MediaConvert Auftrag übernimmt die Transcodierung einer Mediendatei. Wenn Ihr S3-Batchoperationenauftrag Ihre Lambda-Funktion für jedes Video aufruft, erstellt jeder Lambda-Funktionsaufruf einen MediaConvert Transcodierungsauftrag für jedes Eingabevideo.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - MediaConvert Konsole unter https://console.aws.amazon.com/mediaconvert/.
2. Wenn die MediaConvert Einführungsseite angezeigt wird, wählen Sie Erste Schritte aus.
3. Zeige Sie aus der Liste Aufträge jede Zeile an, um die Transcodierungsaufgabe für jedes Eingabe-Video zu überwachen.
4. Geben Sie die Zeile eines Auftrags an, den Sie überprüfen möchten, und wählen Sie den Auftrags-ID-Link, um die Seite Auftrags-Details zu öffnen.
5. Klicken Sie auf der Seite Auftragszusammenfassung unter Ausgaben auf den Link für die Ausgabe HLS, MP4 oder Miniaturansichten, je nachdem, was von Ihrem Browser unterstützt wird, um zum S3-Ziel-Bucket für die Ausgabe-Mediendateien zu wechseln.
6. Wählen Sie im entsprechenden Ordner (HLS, MP4 oder Thumbnails) Ihres S3-Ausgabe-Ziel-Buckets den Namen des Ausgabe-Mediendateiobjekts aus.

   Die Detailseite des Objekts wird geöffnet.
7. Klicken Sie auf der Detailseite des Objekts unter Objekt-Übersichtauf den Link unter Objekt-URL, um die transcodierte Ausgabe-Mediendatei anzusehen.

Schritt 8: Überprüfen Sie die Ausgabe-Mediendateien aus Ihrem S3-Ziel-Bucket

So überprüfen Sie die Ausgabe-Mediendateien aus Ihrem S3-Ziel-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie aus der Liste Buckets den Namen des S3-Ziel-Buckets für Ausgabe-Mediendateien aus, die Sie im Abschnitt Schritt 1 (zum Beispiel **`tutorial-bucket-2`**) erstellt haben.

4. Im Tab Objekte enthält jedes Eingabe-Video einen Ordner mit dem Namen des Eingabe-Videos. Jeder Ordner enthält die transcodierten Ausgabe-Mediendateien für ein Eingabe-Video.

   Gehen Sie wie folgt vor, um die Ausgabe-Mediendateien für ein Eingabe-Video zu überprüfen:

   a. Wählen Sie den Ordner mit dem Namen des Eingabe-Videos, das Sie überprüfen möchten.
   b. Wählen Sie den Ordner Standard/ aus.
   c. Wählen Sie den Ordner für ein transcodiertes Format aus (HLS, MP4 oder Miniaturansichten in diesem Tutorial).
   d. Wählen Sie den Namen der Ausgabemediendatei aus.
   e. Um die transcodierte Datei anzusehen, wählen Sie auf der Detailseite des Objekts den Link unter der Objekt-URL aus.

      Ausgabemediendateien im HLS-Format werden in kurze Segmente aufgeteilt. Um diese Videos abzuspielen, betten Sie die Objekt-URL der `.m3u8`-Datei in einem kompatiblen Player ein.

Schritt 9: Bereinigen

Wenn Sie Videos mit S3 Batch Operations, Lambda und MediaConvert nur als Lernübung transcodiert haben, löschen Sie die AWS Ressourcen, die Sie zugewiesen haben, damit keine Gebühren mehr anfallen.

Teilschritte

- Löschen der S3-Bestandskonfiguration für Ihren S3-Quell-Bucket
- Löschen Sie die Lambda-Funktion
- Löschen der CloudWatch Protokollgruppe
- Löschen Sie die IAM-Rollen zusammen mit den Inline-Richtlinien für die IAM-Rollen
- Löschen Sie die kundenverwaltete IAM-Richtlinie
- Leeren Sie die S3-Buckets
- Löschen der S3-Buckets

Löschen der S3-Bestandskonfiguration für Ihren S3-Quell-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen Ihres Quell-Buckets (z. B. **`tutorial-bucket-1`**).
4. Wählen Sie den Tab Management.
5. Wählen Sie im Abschnitt Bestandskonfigurationen die Optionsschaltfläche neben der Bestandskonfiguration aus, die Sie in Schritt 5 (zum Beispiel **`tutorial-inventory-config`**) erstellt haben.
6. Wählen Sie Löschen und dann Bestätigen aus.

Löschen Sie die Lambda-Funktion

1. Öffnen Sie die - AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie im linken Navigationsbereich die Option Functions (Funktionen) aus.
3. Aktivieren Sie das Kontrollkästchen neben der Funktion, die Sie im Abschnitt Schritt 4 (zum Beispiel **`tutorial-lambda-convert`**) erstellt haben.
4. Wählen Sie Aktionen und anschließend Löschen aus.
5. Wählen Sie im Bestätigungsdialogfeld Delete (Löschen) die Option Delete (Löschen) aus.

Löschen der CloudWatch Protokollgruppe

1. Öffnen Sie die - CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Protokolle und dann Protokollgruppen aus.
3. Aktivieren Sie das Kontrollkästchen neben der Protokollgruppe, deren Name mit der Lambda-Funktion endet, die Sie im Abschnitt Schritt 4 (zum Beispiel **`tutorial-lambda-convert`**) erstellt haben.
4. Wählen Sie Actions (Aktionen) und dann Delete log group(s) (Protokollgruppe(n) löschen) aus.
5. Wählen Sie im Dialogfeld Delete log group(s) (Protokollgruppe(n) löschen) die Option Delete (Löschen) aus.

Löschen Sie die IAM-Rollen zusammen mit den Inline-Richtlinien für die IAM-Rollen

Um die IAM-Rollen zu löschen, die Sie in Schritt 2, Schritt 3, und Schritt 6 erstellt haben, tun Sie Folgendes:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles (Rollen), und aktivieren Sie dann die Kontrollkästchen neben den Rollennamen, die Sie löschen möchten.
3. Wählen Sie oben auf der Seite Delete (Löschen) aus.
4. Geben Sie in das Bestätigungsfeld die erforderliche Antwort basierend auf der Eingabeaufforderung in das Texteingabefeld ein, und wählen Sie dann Löschen aus.

Löschen Sie die kundenverwaltete IAM-Richtlinie

Um die vom Kunden verwaltete IAM-Richtlinie zu löschen, die Sie in Schritt 6 erstellt haben, tun Sie Folgendes:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Richtlinien aus.
3. Wählen Sie das Optionsfeld neben der Richtlinie aus, die Sie in Schritt 6 (zum Beispiel **`tutorial-s3batch-policy`**) erstellt haben. Sie können über das Suchfeld die Liste der Richtlinien filtern.
4. Wählen Sie Aktionen und anschließend Löschen.
5. Bestätigen Sie, dass Sie diese Richtlinie löschen möchten, indem Sie den Namen in das angezeigte Textfeld eingeben und dann Delete (Löschen) wählen.

Leeren Sie die S3-Buckets

Um die S3-Buckets zu leeren, die Sie in Voraussetzungen, Schritt 1 und Schritt 5 erstellt haben, tun Sie Folgendes:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.

3. Wählen Sie in der Liste Buckets das Optionsfeld neben dem Namen des Buckets aus, den Sie leeren möchten. Wählen Sie anschließend Empty (Leeren) aus.
4. Bestätigen Sie auf der Seite Empty bucket (Bucket leeren), dass Sie den Bucket leeren möchten, indem Sie **`permanently delete`** in das Textfeld eingeben und dann Empty (Leeren) auswählen.

Löschen der S3-Buckets

Um die S3-Buckets zu löschen, die Sie in Voraussetzungen, Schritt 1 und Schritt 5 erstellt haben, tun Sie Folgendes:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets das Optionsfeld neben dem Namen des Buckets aus, den Sie löschen möchten.
4. Wählen Sie Löschen aus.
5. Bestätigen Sie auf der Seite Delete bucket (Bucket löschen), dass Sie den Bucket löschen möchten. Geben Sie dazu den Bucket-Namen in das Textfeld ein und wählen Sie Delete bucket (Bucket löschen).

Nächste Schritte

Nachdem Sie dieses Tutorial abgeschlossen haben, können Sie weitere relevante Anwendungsfälle untersuchen:

- Sie können Amazon verwenden CloudFront , um die transcodierten Mediendateien an Zuschauer auf der ganzen Welt zu streamen. Weitere Informationen finden Sie unter Tutorial: Hosten von On-Demand-Streaming-Videos mit Amazon S3 CloudFront, Amazon und Amazon Route 53.
- Sie können Videos in dem Moment transcodieren, indem Sie sie in den S3-Quell-Bucket hochladen. Dazu können Sie einen Amazon S3-Ereignisauslöser konfigurieren, der die Lambda-Funktion automatisch aufruft, um neue Objekte in S3 mit zu transcodieren MediaConvert. Weitere Informationen finden Sie unter Tutorial: Verwenden eines Amazon S3-Auslösers zum Aufrufen einer Lambda-Funktion im AWS Lambda -Entwicklerhandbuch.

Tutorial: Konfigurieren einer statischen Website auf Amazon S3

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Sie können einen Amazon S3-Bucket so konfigurieren, dass er sich wie eine Website verhält. In diesem Beispiel sehen Sie die Schritte für das Hosten einer Website auf Amazon S3.

> **Important**
>
> Für das folgende Tutorial ist die Deaktivierung des öffentlichen Zugriffs erforderlich. Wir empfehlen, alle Einstellungen für das Blockieren des öffentlichen Zugriffs aktiviert zu lassen. Wenn Sie alle vier Block Public Access-Einstellungen aktiviert lassen und eine statische Website hosten möchten, können Sie die Amazon- CloudFront Ursprungszugriffssteuerung (OAC) verwenden. Amazon CloudFront bietet die Funktionen, die zum Einrichten einer sicheren statischen Website erforderlich sind. Statische Amazon-S3-Websites unterstützen nur HTTP-Endpunkte. Amazon CloudFront verwendet den dauerhaften Speicher von Amazon S3 und bietet gleichzeitig zusätzliche Sicherheitsheader wie HTTPS. HTTPS erhöht die Sicherheit, indem eine normale HTTP-Anforderung verschlüsselt und vor gängigen Cyberangriffen geschützt wird. Weitere Informationen finden Sie unter Erste Schritte mit einer sicheren statischen Website im Amazon- CloudFront Entwicklerhandbuch.

Themen

Schritt 1: Erstellen eines Buckets

Die folgenden Anweisungen geben einen Überblick darüber, wie Sie Ihre Buckets für das Website-Hosting erstellen. Detaillierte step-by-step Anweisungen zum Erstellen eines Buckets finden Sie unter Erstellen eines Buckets.

So erstellen Sie einen Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie Create Bucket (Bucket erstellen).
3. Geben Sie den Bucket-Namen ein (z. B. **`example.com`**).
4. Wählen Sie die Region aus, in der Sie Ihren Bucket erstellen möchten.

   Wählen Sie eine Region in Ihrer Nähe aus, um Latenz und Kosten auf einem Minimum zu halten oder behördliche Vorschriften zu erfüllen. Die von Ihnen ausgewählte Region bestimmt Ihren Amazon-S3-Website-Endpunkt. Weitere Informationen finden Sie unter Website-Endpunkte.
5. Um die Standardeinstellungen zu übernehmen und den Bucket zu erstellen, wählen Sie Create (Erstellen).

Schritt 2: Aktivieren des statischen Website-Hostings

Nach der Erstellung eines Buckets können Sie das statische Website-Hosting für Ihren Bucket aktivieren. Sie können einen neuen Bucket erstellen oder einen vorhandenen Bucket verwenden.

So aktivieren Sie das statische Website-Hosting

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie das Hosting statischer Websites aktivieren wollen.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Static website hosting (Hosting statischer Websites) Edit (Bearbeiten) aus.
5. Wählen Sie Use this bucket to host a website (Diesen Bucket zum Hosten einer Website verwenden).
6. Wählen Sie unter Static website hosting (Hosting statischer Websites) die Option Enable (Aktivieren) aus
7. Geben Sie unter Index document (Index-Dokument) den Dateinamen des Index-Dokuments ein, der typischerweise `index.html` ist.

   Der Name des Indexdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Indexdokuments übereinstimmen, das Sie in den S3-Bucket hochladen möchten. Wenn Sie Ihren Bucket für das Hosting von Websites konfigurieren, müssen Sie ein Indexdokument angeben. Amazon S3 gibt dieses Indexdokument zurück, wenn Anfragen an die Root-Domäne oder einen der Unterordner gestellt werden. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments.
8. Um ein eigenes benutzerdefiniertes Fehlerdokument für Fehler der Klasse 4XX bereitzustellen, geben Sie unter Fehlerdokument den Dateinamen des benutzerdefinierten Fehlerdokuments ein.

   Der Name des Fehlerdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Fehlerdokuments übereinstimmen, das Sie in Ihren S3-Bucket hochladen möchten. Wenn Sie kein benutzerdefiniertes Fehlerdokument angeben und ein Fehler auftritt, wird von Amazon S3 ein Standard-HTML-Fehlerdokument zurückgegeben. Weitere Informationen finden Sie unter Konfigurieren eines benutzerdefinierten Fehlerdokuments.
9. (Optional) Wenn Sie fortschrittliche Umleitungsregeln angeben möchten, geben Sie unter Redirection rules (Umleitungsregeln) JSON zur Beschreibung der Regeln ein.

   Beispielsweise können Sie bedingt Anfragen abhängig von bestimmten Objektschlüsselnamen oder Präfixen in der Anfrage weiterleiten. Weitere Informationen finden Sie unter Konfigurieren von Umleitungsregeln für die Verwendung von fortschrittliche bedingten Umleitungen.
10. Wählen Sie Save Changes (Änderungen speichern).

    Amazon S3 ermöglicht statisches Website-Hosting für Ihren Bucket. Unten auf der Seite sehen Sie unter Static website hosting (Hosting statischer Websites) den Website-Endpunkt für Ihren Bucket.

11. Notieren Sie unter Static website hosting (Statisches Website-Hosting) den Wert für Endpoint (Endpunkt).

    Der Endpoint (Endpunkt) ist der Amazon-S3-Website-Endpunkt für Ihren Bucket. Nachdem Sie den Bucket als statische Website konfiguriert haben, können Sie diesen Endpunkt verwenden, um Ihre Website zu testen.

Schritt 3: Bearbeiten der Block-Public-Access-Einstellungen

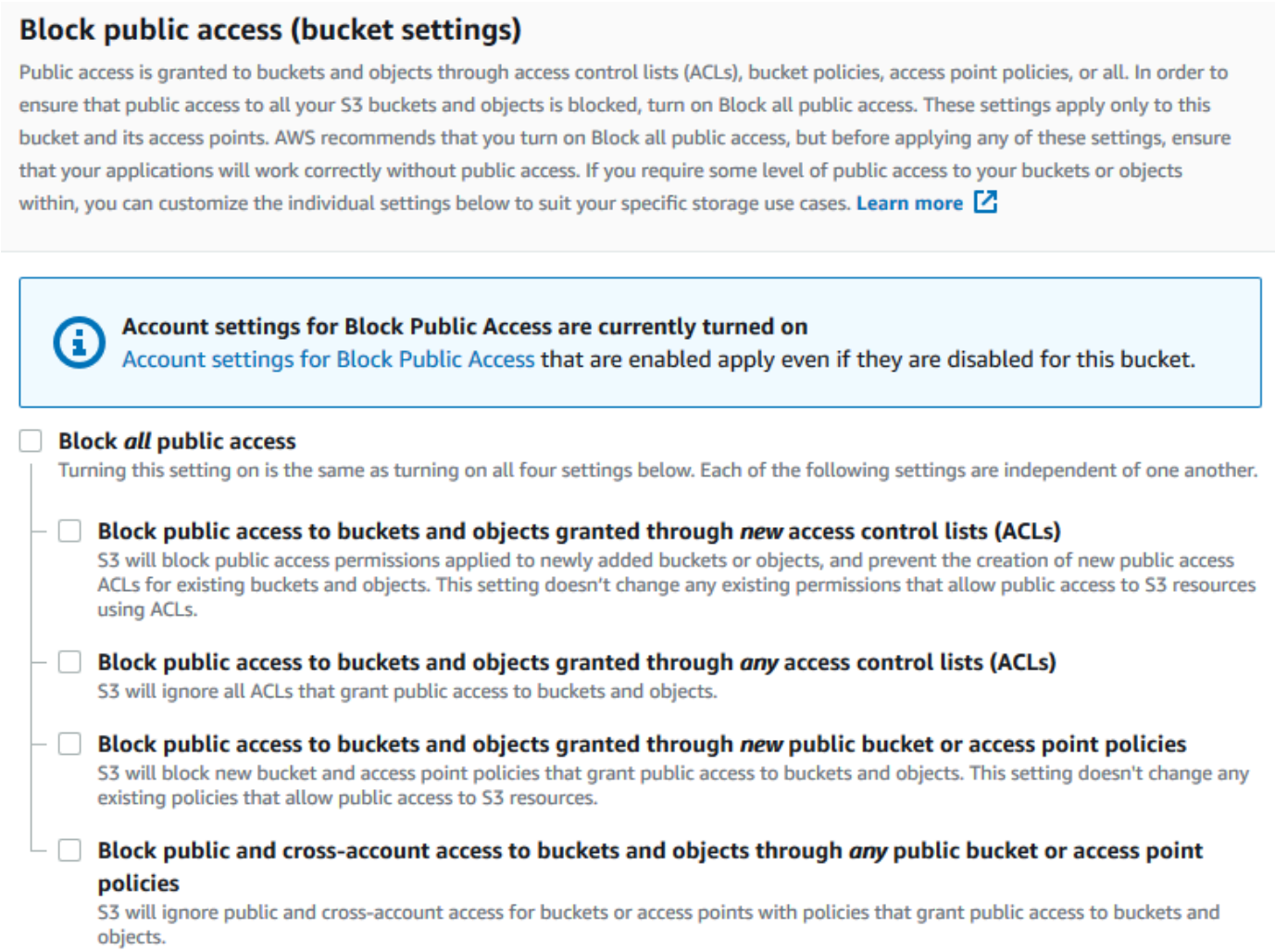
Standardmäßig blockiert Amazon S3 den öffentlichen Zugriff auf Ihr Konto und Ihre Buckets. Wenn Sie einen Bucket verwenden möchten, um eine statische Website zu hosten, können Sie diese Schritte verwenden, um Ihre Einstellungen für Block Public Access zu bearbeiten:

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Permissions (Berechtigungen).
4. Wählen Sie unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)), die Option Edit (Bearbeiten).
5. Löschen Sie Block all public access (Gesamten öffentlichen Zugriff blockieren) und wählen Sie Save (Speichern).

   > **Warning**
   >
   > Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich

zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.



Amazon S3 deaktiviert die Block Public Access-Einstellungen für Ihren Bucket. Um eine öffentliche, statische Website zu erstellen, müssen Sie möglicherweise auch die Block Public Access-Einstellungen für Ihr Konto bearbeiten, bevor Sie eine Bucket-Richtlinie hinzufügen. Wenn Kontoeinstellungen für Block Public Access derzeit aktiviert sind, wird unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)) ein Hinweis angezeigt.

Schritt 4: Hinzufügen einer Bucket-Richtlinie, die den Inhalt Ihres Buckets öffentlich verfügbar macht

Nachdem Sie die Einstellungen für S3 Block Public Access bearbeitet haben, können Sie eine Bucket-Richtlinie hinzufügen, um öffentlichen Lesezugriff auf den Bucket zu gewähren. Wenn Sie öffentlichen Lesezugriff gewähren, kann jeder im Internet auf Ihren Bucket zugreifen.

> **Important**
>
> Die zuvor genannte Richtlinie ist nur ein Beispiel und erlaubt Vollzugriff auf die Inhalte Ihres Buckets. Bevor Sie mit diesem Schritt fortfahren, lesen Sie den Abschnitt Wie kann ich die Dateien in meinem Amazon-S3-Bucket sichern?, um sicherzustellen, dass Sie die bewährten Methoden zum Sichern der Dateien in Ihrem S3-Bucket und die Risiken in Zusammenhang mit der Gewährung von öffentlichem Zugriff kennen.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Permissions (Berechtigungen).
3. Wählen Sie unter Bucket Policy (Bucket-Richtlinie) Edit (Bearbeiten).
4. Um öffentlichen Lesezugriff auf Ihre Website zu gewähren, kopieren Sie die folgende Bucket-Richtlinie und fügen Sie sie in den Bucket policy editor (Bucket-Richtlinieneditor) ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::Bucket-Name/*"
            ]
        }
    ]
}
```

5. Aktualisieren Sie den `Resource` zu Ihrem Bucket-Namen.

   In der obigen Beispiel-Bucket-Richtlinie ist *`Bucket-Name`* ein Platzhalter für den Bucket-Namen. Um diese Bucket-Richtlinie mit Ihrem eigenen Bucket zu verwenden, müssen Sie diesen Namen so aktualisieren, dass er mit Ihrem Bucket übereinstimmt.
6. Wählen Sie Save Changes (Änderungen speichern).

   Es wird eine Meldung angezeigt, die darauf hinweist, dass die Bucket-Richtlinie erfolgreich hinzugefügt wurde.

   Wenn die Fehlermeldung `Policy has invalid resource` angezeigt wird, bestätigen Sie, dass der Bucket-Name in der Bucket-Richtlinie mit Ihrem Bucket-Namen übereinstimmt. Informationen zum Hinzufügen einer Bucket-Richtlinie finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu?

   Wenn Sie eine Fehlermeldung erhalten und die Bucket-Richtlinie nicht speichern können, überprüfen Sie Ihr Konto und die Bucket-Einstellungen für Block Public Access, um zu bestätigen, dass Sie den öffentlichen Zugriff auf den Bucket zulassen.

Schritt 5: Konfigurieren eines Indexdokuments

Wenn Sie das statische Website-Hosting für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B, **`index.html`**). Nachdem Sie das Hosting statischer Websites für den Bucket aktiviert haben, laden Sie eine HTML-Datei mit diesem Indexdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie das Indexdokument

1. Erstellen Sie eine Datei `index.html`.

   Wenn Sie nicht über eine Datei `index.html` verfügen, können Sie mit dem folgenden HTML-Code eine Datei erstellen:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>My Website Home Page</title>
</head>
<body>
  <h1>Welcome to my website</h1>
  <p>Now hosted on Amazon S3!</p>
```

```
</body>
</html>
```

2. Speichern Sie die Indexdatei lokal.

   Der Dateiname des Indexdokuments muss genau mit dem Namen des Indexdokuments übereinstimmen, den Sie im Dialogfeld Static website hosting (Statisches Website-Hosting) eingeben. Beim Namen des Indexdokuments wird die Groß- und Kleinschreibung berücksichtigt. Wenn Sie beispielsweise im Dialogfeld Static website hosting (Statisches Website-Hosting) `index.html` als den Namen des Index document (Indexdokuments) eingeben, muss der Dateiname des Indexdokuments ebenfalls `index.html` und nicht `Index.html` lauten.

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.

5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Indexdokuments ein (z. B. `index.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings.

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.

6. Führen Sie einen der folgenden Schritte aus, um das Indexdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Indexdatei per Drag & Drop in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   step-by-step Anweisungen finden Sie unter Objekte hochladen.

7. (Optional) Laden Sie andere Website-Inhalte in Ihren Bucket hoch.

Schritt 6: Konfigurieren eines Fehlerdokuments

Wenn Sie das Hosting statischer Websites für Ihren Bucket aktivieren, geben Sie den Namen des Fehlerdokuments ein (z. B, **`404.html`**). Nachdem das Hosting statischer Websites für den Bucket aktiviert wurde, laden Sie eine HTML-Datei mit diesem Fehlerdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie ein Fehlerdokument

1. Erstellen Sie ein Fehlerdokument, z. B. `404.html`.
2. Speichern Sie die Fehlerdokumentdatei lokal.

   Der Name des Fehlerdokuments unterscheidet zwischen Groß- und Kleinschreibung und muss genau mit dem Namen übereinstimmen, den Sie beim Aktivieren des statischen Website-Hostings eingeben. Wenn Sie beispielsweise `404.html` im Dialogfeld Hosten einer statischen Website als Namen des Fehlerdokuments eingeben, muss der Dateiname des Fehlerdokuments ebenfalls `404.html` lauten.
3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.
5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Fehlerdokuments ein (z. B, `404.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings und Konfigurieren eines benutzerdefinierten Fehlerdokuments.

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.
6. Führen Sie einen der folgenden Schritte aus, um das Fehlerdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Fehlerdokumentdatei in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   step-by-step Anweisungen finden Sie unter Objekte hochladen.

Schritt 7: Testen des Website-Endpunkts

Nach der Konfigurierung des statischen Website-Hostings für Ihren Bucket können Sie den Website-Endpunkt testen.

> **Note**
>
> Amazon S3 unterstützt keinen HTTPS-Zugriff auf die Website. Wenn Sie HTTPS verwenden möchten, können Sie Amazon verwenden, CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird.
> Weitere Informationen finden Sie unter Wie verwende ich , CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird? und Erzwingen von HTTPS für die Kommunikation zwischen Viewern und CloudFront.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Properties (Eigenschaften).
3. Wählen Sie unten auf der Seite unter Static website hosting (Hosting statischer Websites) Ihren Bucket-Website-Endpunkt.

   Ihr Indexdokument wird in einem separaten Browserfenster geöffnet.

Sie haben jetzt eine auf Amazon S3 gehostete Website. Diese Website steht am Amazon S3-Website-Endpunkt zur Verfügung. Möglicherweise haben Sie jedoch eine Domäne wie `example.com`, die den Inhalt von der von Ihnen erstellten Website bereitstellen soll. Sie könnten auch die Amazon S3-Unterstützung der Root-Domäne nutzen, um Anforderungen für `http://www.example.com` und `http://example.com` zu bedienen. Dafür sind zusätzliche Schritte erforderlich. Ein Beispiel finden Sie unter Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne.

Schritt 8: Bereinigen

Wenn Sie die statische Website nur zur Übung erstellt haben, löschen Sie die AWS -Ressourcen, die Sie zugewiesen haben, damit keine weiteren Kosten für Sie anfallen. Nachdem Sie Ihre AWS Ressourcen gelöscht haben, ist Ihre Website nicht mehr verfügbar. Weitere Informationen finden Sie unter Löschen eines Buckets.

Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne

Angenommen, Sie wollen eine statische Website auf Amazon S3 hosten. Sie haben eine Domäne bei Amazon Route 53 registriert (zum Beispiel `example.com`), und Sie möchten, dass Anforderungen für `http://www.example.com` und `http://example.com` von Ihren Amazon-S3-Inhalten aus bedient werden. Sie können diese Anleitung verwenden, um zu erfahren, wie Sie eine statische Website hosten und Umleitungen auf Amazon S3 für eine Website mit einem benutzerdefinierten Domänennamen, der bei Route 53 registriert ist, anlegen können. Sie können mit einer vorhandenen Website arbeiten, die Sie auf Amazon S3 hosten möchten, oder diesen Walkthrough verwenden, um bei Null anzufangen.

Nach Abschluss dieser Anleitung können Sie optional Amazon verwenden, CloudFront um die Leistung Ihrer Website zu verbessern. Weitere Informationen finden Sie unter Beschleunigen Ihrer Website mit Amazon CloudFront.

> **Note**
>
> Amazon S3 Website-Endpunkte unterstützen nicht HTTPS oder Zugriffspunkte. Wenn Sie HTTPS verwenden möchten, können Sie Amazon verwenden, CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird.
> Weitere Informationen finden Sie unter Wie verwende ich , CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird? und Erzwingen von HTTPS für die Kommunikation zwischen Viewern und CloudFront.

Automatisieren der statischen Website-Einrichtung mit einer - AWS CloudFormation Vorlage

Sie können eine - AWS CloudFormation Vorlage verwenden, um Ihre statische Website-Einrichtung zu automatisieren. Die AWS CloudFormation Vorlage richtet die Komponenten ein, die Sie zum Hosten einer sicheren statischen Website benötigen, sodass Sie sich mehr auf den Inhalt Ihrer Website und weniger auf die Konfiguration von Komponenten konzentrieren können.

Die AWS CloudFormation Vorlage enthält die folgenden Komponenten:

- Amazon S3 – Erstellt einen Amazon-S3-Bucket zum Hosten Ihrer statischen Website.
- CloudFront – Erstellt eine CloudFront Verteilung, um Ihre statische Website zu beschleunigen.

- Lambda@Edge – Verwendet Lambda@Edge, um jeder Serverantwort Sicherheits-Header hinzuzufügen. Sicherheits-Header sind eine Gruppe von Headern in der Webserverantwort, die Webbrowser anweisen, zusätzliche Sicherheitsvorkehrungen zu treffen. Weitere Informationen finden Sie im Blogbeitrag Hinzufügen von HTTP-Sicherheitsheadern mit Lambda@Edge und Amazon CloudFront.

Diese AWS CloudFormation Vorlage steht Ihnen zum Herunterladen und Verwenden zur Verfügung. Informationen und Anweisungen finden Sie unter Erste Schritte mit einer sicheren statischen Website im Amazon- CloudFront Entwicklerhandbuch.

Themen

Bevor Sie beginnen

Während Sie den Schritten in diesem Beispiel folgen, arbeiten Sie mit den folgenden Services:

Amazon Route 53 – Sie verwenden Route 53, um Domänen zu registrieren und zu definieren, wohin der Internetdatenverkehr für Ihre Domäne geleitet werden soll. Das Beispiel zeigt, wie Sie Route 53-Alias-Datensätze erstellen, mit denen Sie Datenverkehr für Ihre Domäne (`example.com`)

und Subdomäne (`www.example.com`) in einen Amazon-S3-Bucket umleiten, der eine HTML-Datei enthält.

Amazon S3 – Mit Amazon S3 erstellen Sie Buckets, laden eine Beispielseite für eine Website hoch, konfigurieren Berechtigungen, sodass jeder den Inhalt sehen kann, und konfigurieren dann die Buckets für das Website-Hosting.

Schritt 1: Registrieren einer benutzerdefinierten Domäne bei Route 53

Wenn Sie noch keinen registrierten Domänennamen haben, wie z. B. `example.com`, registrieren Sie einen mit Route 53. Weitere Informationen finden Sie unter Registrieren einer neuen Domäne im Amazon-Route 53-Entwicklerhandbuch. Nachdem Sie Ihren Domänennamen registriert haben, können Sie Ihre Amazon-S3-Buckets für das Website-Hosting erstellen und konfigurieren.

Schritt 2: Erstellen von zwei Buckets

Um Anforderungen von sowohl der Stammdomäne als auch von der Subdomäne zu unterstützen, erstellen Sie zwei Buckets.

- Domänen-Bucket – `example.com`
- Subdomänen-Bucket – `www.example.com`

Diese Bucket-Namen müssen genau mit Ihrem Domänennamen übereinstimmen. In diesem Beispiel lautet der Domänenname `example.com`. Sie hosten Ihre Inhalte aus dem Stammdomänen-Bucket (`example.com`). Sie erstellen eine Umleitungsanforderung für den Subdomänen-Bucket (`www.example.com`). Falls ein Benutzer `www.example.com` in seinen Browser eingibt, wird er zu `example.com` umgeleitet und sieht den Inhalt, der im Amazon-S3-Bucket mit diesem Namen gehostet wird.

So erstellen Sie Ihre Buckets für Website-Hosting

Die folgenden Anweisungen geben einen Überblick darüber, wie Sie Ihre Buckets für das Website-Hosting erstellen. Detaillierte step-by-step Anweisungen zum Erstellen eines Buckets finden Sie unter Erstellen eines Buckets.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Erstellen Sie den Stammdomänen-Bucket:

a. Wählen Sie Create Bucket (Bucket erstellen) aus.

b. Geben Sie den Bucket-Namen ein (z. B. **`example.com`**).

c. Wählen Sie die Region aus, in der Sie Ihren Bucket erstellen möchten.

   Wählen Sie eine Region in Ihrer Nähe aus, um Latenz und Kosten auf einem Minimum zu halten oder behördliche Vorschriften zu erfüllen. Die von Ihnen ausgewählte Region bestimmt Ihren Amazon-S3-Website-Endpunkt. Weitere Informationen finden Sie unter Website-Endpunkte.

d. Um die Standardeinstellungen zu übernehmen und den Bucket zu erstellen, wählen Sie Create (Erstellen).

3. Erstellen Sie Ihren Subdomänen-Bucket:

a. Wählen Sie Create Bucket (Bucket erstellen) aus.

b. Geben Sie den Bucket-Namen ein (z. B. **`www.example.com`**).

c. Wählen Sie die Region aus, in der Sie Ihren Bucket erstellen möchten.

   Wählen Sie eine Region in Ihrer Nähe aus, um Latenz und Kosten auf einem Minimum zu halten oder behördliche Vorschriften zu erfüllen. Die von Ihnen ausgewählte Region bestimmt Ihren Amazon-S3-Website-Endpunkt. Weitere Informationen finden Sie unter Website-Endpunkte.

d. Um die Standardeinstellungen zu übernehmen und den Bucket zu erstellen, wählen Sie Create (Erstellen).

Im nächsten Schritt konfigurieren Sie `example.com` für das Website-Hosting.

Schritt 3: Konfigurieren Ihres Stammdomänen-Buckets für Website-Hosting

In diesem Schritt konfigurieren Sie Ihren Stammdomänen-Bucket (`example.com`) als Website. Dieser Bucket enthält Ihren Website-Inhalt. Wenn Sie einen Bucket für das Website-Hosting konfigurieren, können Sie über auf die Website zugreife Website-Endpunkte.

So aktivieren Sie das statische Website-Hosting

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie das Hosting statischer Websites aktivieren wollen.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Static website hosting (Hosting statischer Websites) Edit (Bearbeiten) aus.
5. Wählen Sie Use this bucket to host a website (Diesen Bucket zum Hosten einer Website verwenden).
6. Wählen Sie unter Static website hosting (Hosting statischer Websites) die Option Enable (Aktivieren) aus
7. Geben Sie unter Index document (Index-Dokument) den Dateinamen des Index-Dokuments ein, der typischerweise `index.html` ist.

   Der Name des Indexdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Indexdokuments übereinstimmen, das Sie in den S3-Bucket hochladen möchten. Wenn Sie Ihren Bucket für das Hosting von Websites konfigurieren, müssen Sie ein Indexdokument angeben. Amazon S3 gibt dieses Indexdokument zurück, wenn Anfragen an die Root-Domäne oder einen der Unterordner gestellt werden. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments.
8. Um ein eigenes benutzerdefiniertes Fehlerdokument für Fehler der Klasse 4XX bereitzustellen, geben Sie unter Fehlerdokument den Dateinamen des benutzerdefinierten Fehlerdokuments ein.

   Der Name des Fehlerdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Fehlerdokuments übereinstimmen, das Sie in Ihren S3-Bucket hochladen möchten. Wenn Sie kein benutzerdefiniertes Fehlerdokument angeben und ein Fehler auftritt, wird von Amazon S3 ein Standard-HTML-Fehlerdokument zurückgegeben. Weitere Informationen finden Sie unter Konfigurieren eines benutzerdefinierten Fehlerdokuments.
9. (Optional) Wenn Sie fortschrittliche Umleitungsregeln angeben möchten, geben Sie unter Redirection rules (Umleitungsregeln) JSON zur Beschreibung der Regeln ein.

   Beispielsweise können Sie bedingt Anfragen abhängig von bestimmten Objektschlüsselnamen oder Präfixen in der Anfrage weiterleiten. Weitere Informationen finden Sie unter Konfigurieren von Umleitungsregeln für die Verwendung von fortschrittliche bedingten Umleitungen.
10. Wählen Sie Save Changes (Änderungen speichern).

    Amazon S3 ermöglicht statisches Website-Hosting für Ihren Bucket. Unten auf der Seite sehen Sie unter Static website hosting (Hosting statischer Websites) den Website-Endpunkt für Ihren Bucket.

11. Notieren Sie unter Static website hosting (Statisches Website-Hosting) den Wert für Endpoint (Endpunkt).

    Der Endpoint (Endpunkt) ist der Amazon-S3-Website-Endpunkt für Ihren Bucket. Nachdem Sie den Bucket als statische Website konfiguriert haben, können Sie diesen Endpunkt verwenden, um Ihre Website zu testen.

Nachdem Sie die Blockierungseinstellungen für den öffentlichen Zugriff bearbeitet und eine Bucket-Richtlinie hinzugefügt haben, die öffentlichen Lesezugriff ermöglicht, können Sie den Website-Endpunkt verwenden, um auf Ihre Website zuzugreifen.

Im nächsten Schritt konfigurieren Sie Ihre Subdomäne (`www.example.com`) zur Weiterleitung von Anforderungen an Ihre Domäne (`example.com`).

Schritt 4: Konfigurieren Ihres Subdomänen-Buckets für die Website-Umleitung

Nachdem Sie Ihren Stammdomänen-Bucket für das Website-Hosting konfiguriert haben, können Sie Ihren Subdomänen-Bucket so konfigurieren, dass alle Anforderungen zu der Domäne umgeleitet werden. In diesem Beispiel werden alle Anforderungen für `www.example.com` an `example.com` umgeleitet.

So konfigurieren Sie eine Umleitungsanforderung

1. Wählen Sie in der Amazon-S3-Konsole in der Liste Buckets Ihren Subdomänen-Bucket aus (in diesem Beispiel `www.example.com`).
2. Wählen Sie Properties (Eigenschaften).
3. Wählen Sie unter Static website hosting (Hosting statischer Websites) Edit (Bearbeiten) aus.
4. Wählen Sie Redirect requests for an object (Anfragen für ein Objekt umleiten).
5. Geben Sie im Feld Target bucket (Ziel-Bucket) Ihre Root-Domäne ein, z. B. **`example.com`**.
6. Wählen Sie für Protocol (Protokoll) die Option http aus.
7. Wählen Sie Save Changes (Änderungen speichern).

Schritt 5: Konfigurieren der Protokollierung für Website-Datenverkehr

Wenn Sie die Anzahl der Besucher nachverfolgen möchten, die auf Ihre Website zugreifen, können Sie optional die Protokollierung für Ihren Stammdomänen-Bucket aktivieren. Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung. Wenn Sie Amazon verwenden möchten, CloudFront um Ihre Website zu beschleunigen, können Sie auch die - CloudFront Protokollierung verwenden.

So aktivieren Sie die Server-Zugriffsprotokollierung für Ihren Stammdomänen-Bucket

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Erstellen Sie in derselben Region, in der Sie den Bucket erstellt haben, der als statische Website konfiguriert ist, einen Bucket für die Protokollierung, z. B. `logs.example.com`.
3. Erstellen Sie einen Ordner für die Protokolldateien der Server-Zugriffsprotokollierung (z. B, `logs`).
4. (Optional) Wenn Sie verwenden möchten, um die Leistung Ihrer Website CloudFront zu verbessern, erstellen Sie einen Ordner für die CloudFront Protokolldateien (z. B. `cdn`).

   > **Important**
   >
   > Wenn Sie eine Verteilung erstellen oder aktualisieren und die CloudFront Protokollierung aktivieren, CloudFront aktualisiert die Bucket-Zugriffssteuerungsliste (ACL), um dem `awslogsdelivery` Konto `FULL_CONTROL` Berechtigungen zum Schreiben von Protokollen in Ihren Bucket zu erteilen. Weitere Informationen finden Sie unter Erforderliche Berechtigungen zum Konfigurieren der Standardprotokollierung und zum Zugriff auf Ihre Protokolldateien im Amazon- CloudFront Entwicklerhandbuch. Wenn der Bucket, der die Protokolle speichert, die Einstellung „Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, um ACLs zu deaktivieren, CloudFront kann keine Protokolle in den Bucket schreiben. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..
5. Wählen Sie in der Liste Buckets Ihren Stammdomänen-Bucket aus.
6. Wählen Sie Properties (Eigenschaften).
7. Wählen Sie unter Server access logging (Server-Zugriffsprotokollierung) Edit (Bearbeiten).
8. Wählen Sie Enable aus.

9. Wählen Sie im Ziel-Bucket den Bucket und das Ordnerziel für die Server-Zugriffsprotokolle aus:

   - Navigieren Sie zum Ordner- und Bucket-Speicherort:
     1. Wählen Sie Browse S3 (S3 durchsuchen).
     2. Wählen Sie den Bucket-Namen und dann den Ordner mit den Protokollen aus.
     3. Wählen Sie Choose path (Pfad wählen).
   - Geben Sie den S3-Bucket-Pfad ein, z. B, `s3://logs.example.com/logs/`.

10. Wählen Sie Save Changes (Änderungen speichern).

    In Ihrem Protokoll-Bucket können Sie jetzt auf Ihre Protokolle zugreifen. Amazon S3 schreibt Website-Zugriffsprotokolle alle zwei Stunden in Ihren Bucket zur Protokollierung.

Schritt 6: Hochladen des Index und des Website-Inhalts

Als Nächstes laden Sie das Indexdokument und den optionalen Website-Inhalt zum Stammdomänen-Bucket hoch.

Wenn Sie das statische Website-Hosting für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B, **`index.html`**). Nachdem Sie das Hosting statischer Websites für den Bucket aktiviert haben, laden Sie eine HTML-Datei mit diesem Indexdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie das Indexdokument

1. Erstellen Sie eine Datei `index.html`.

   Wenn Sie nicht über eine Datei `index.html` verfügen, können Sie mit dem folgenden HTML-Code eine Datei erstellen:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>My Website Home Page</title>
</head>
<body>
  <h1>Welcome to my website</h1>
  <p>Now hosted on Amazon S3!</p>
</body>
</html>
```

2. Speichern Sie die Indexdatei lokal.

   Der Dateiname des Indexdokuments muss genau mit dem Namen des Indexdokuments übereinstimmen, den Sie im Dialogfeld Static website hosting (Statisches Website-Hosting) eingeben. Beim Namen des Indexdokuments wird die Groß- und Kleinschreibung berücksichtigt. Wenn Sie beispielsweise im Dialogfeld Static website hosting (Statisches Website-Hosting) `index.html` als den Namen des Index document (Indexdokuments) eingeben, muss der Dateiname des Indexdokuments ebenfalls `index.html` und nicht `Index.html` lauten.

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.

5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Indexdokuments ein (z. B. `index.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings.

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.

6. Führen Sie einen der folgenden Schritte aus, um das Indexdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Indexdatei per Drag & Drop in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   step-by-step Anweisungen finden Sie unter Objekte hochladen.

7. (Optional) Laden Sie andere Website-Inhalte in Ihren Bucket hoch.

Schritt 7: Hochladen eines Fehlerdokuments

Wenn Sie das Hosting statischer Websites für Ihren Bucket aktivieren, geben Sie den Namen des Fehlerdokuments ein (z. B, **`404.html`**). Nachdem das Hosting statischer Websites für den Bucket aktiviert wurde, laden Sie eine HTML-Datei mit diesem Fehlerdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie ein Fehlerdokument

1. Erstellen Sie ein Fehlerdokument, z. B. `404.html`.

2. Speichern Sie die Fehlerdokumentdatei lokal.

   Der Name des Fehlerdokuments unterscheidet zwischen Groß- und Kleinschreibung und muss genau mit dem Namen übereinstimmen, den Sie beim Aktivieren des statischen Website-Hostings eingeben. Wenn Sie beispielsweise `404.html` im Dialogfeld Hosten einer statischen Website als Namen des Fehlerdokuments eingeben, muss der Dateiname des Fehlerdokuments ebenfalls `404.html` lauten.

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.

5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Fehlerdokuments ein (z. B, `404.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings und Konfigurieren eines benutzerdefinierten Fehlerdokuments.

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.

6. Führen Sie einen der folgenden Schritte aus, um das Fehlerdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Fehlerdokumentdatei in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   step-by-step Anweisungen finden Sie unter Objekte hochladen.

Schritt 8: Bearbeiten der S3 Block Public Access-Einstellungen

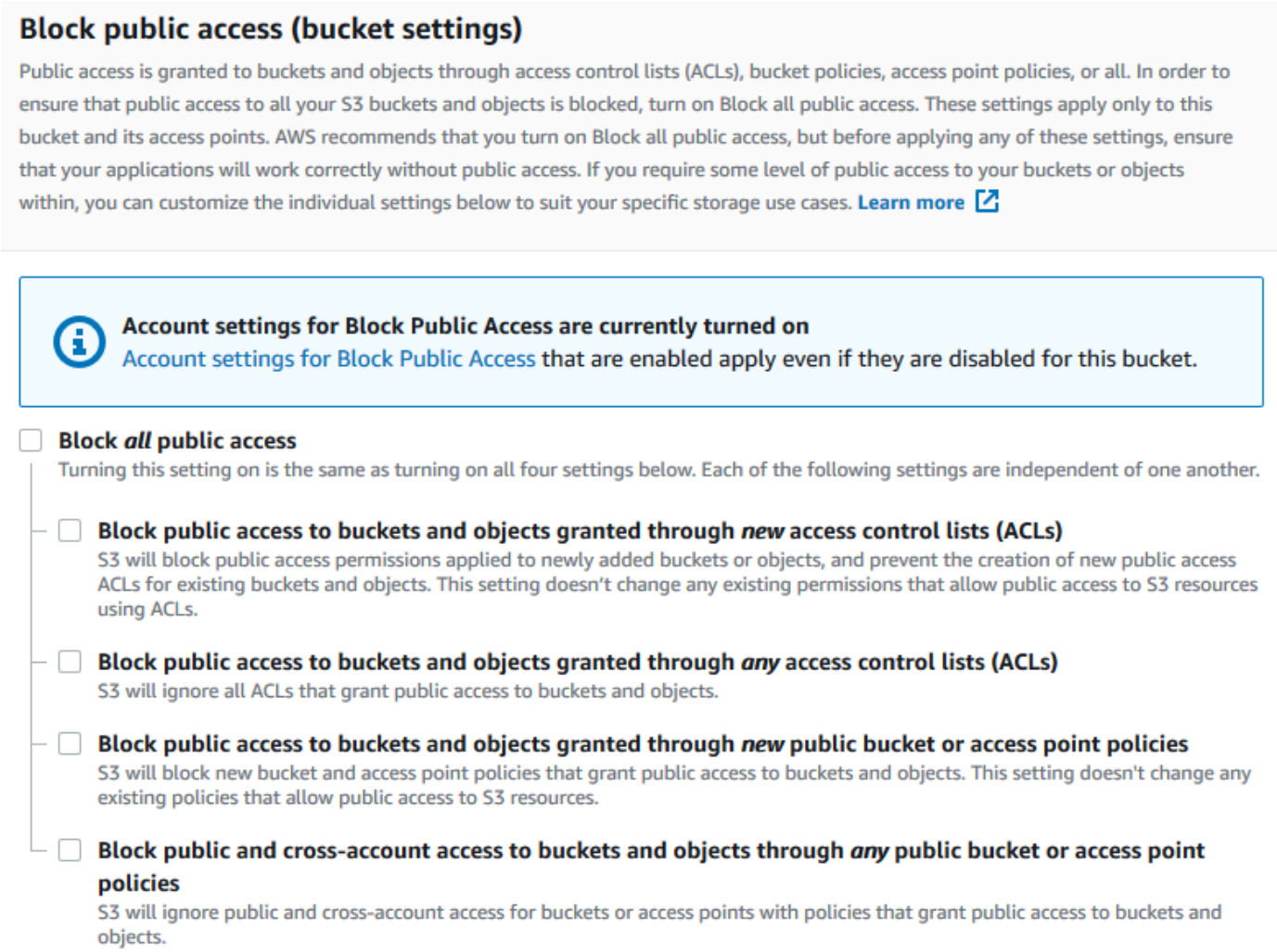
In diesem Beispiel bearbeiten Sie die Block-Einstellungen für den öffentlichen Zugriff für den Domänen-Bucket (`example.com`), um den öffentlichen Zugriff zu ermöglichen.

Standardmäßig blockiert Amazon S3 den öffentlichen Zugriff auf Ihr Konto und Ihre Buckets. Wenn Sie einen Bucket verwenden möchten, um eine statische Website zu hosten, können Sie diese Schritte verwenden, um Ihre Einstellungen für Block Public Access zu bearbeiten:

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Permissions (Berechtigungen).
4. Wählen Sie unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)), die Option Edit (Bearbeiten).
5. Löschen Sie Block all public access (Gesamten öffentlichen Zugriff blockieren) und wählen Sie Save (Speichern).

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.



Amazon S3 deaktiviert die Block Public Access-Einstellungen für Ihren Bucket. Um eine öffentliche, statische Website zu erstellen, müssen Sie möglicherweise auch die Block Public Access-Einstellungen für Ihr Konto bearbeiten, bevor Sie eine Bucket-Richtlinie hinzufügen. Wenn Kontoeinstellungen für Block Public Access derzeit aktiviert sind, wird unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)) ein Hinweis angezeigt.

Schritt 9: Anfügen einer Bucket-Richtlinie

In diesem Beispiel hängen Sie eine Bucket-Richtlinie an den Domain-Bucket (`example.com`) an, um öffentlichen Lesezugriff zu ermöglichen. Sie ersetzen beispielsweise den *`Bucket-Namen`* in der Beispiel-Bucket-Richtlinie durch den Namen Ihres Domain-Buckets `example.com`.

Nachdem Sie die Einstellungen für S3 Block Public Access bearbeitet haben, können Sie eine Bucket-Richtlinie hinzufügen, um öffentlichen Lesezugriff auf den Bucket zu gewähren. Wenn Sie öffentlichen Lesezugriff gewähren, kann jeder im Internet auf Ihren Bucket zugreifen.

> **Important**
>
> Die zuvor genannte Richtlinie ist nur ein Beispiel und erlaubt Vollzugriff auf die Inhalte Ihres Buckets. Bevor Sie mit diesem Schritt fortfahren, lesen Sie den Abschnitt Wie kann ich die Dateien in meinem Amazon-S3-Bucket sichern?, um sicherzustellen, dass Sie die bewährten Methoden zum Sichern der Dateien in Ihrem S3-Bucket und die Risiken in Zusammenhang mit der Gewährung von öffentlichem Zugriff kennen.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Permissions (Berechtigungen).
3. Wählen Sie unter Bucket Policy (Bucket-Richtlinie) Edit (Bearbeiten).
4. Um öffentlichen Lesezugriff auf Ihre Website zu gewähren, kopieren Sie die folgende Bucket-Richtlinie und fügen Sie sie in den Bucket policy editor (Bucket-Richtlinieneditor) ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::Bucket-Name/*"
            ]
        }
    ]
}
```

5. Aktualisieren Sie den `Resource` zu Ihrem Bucket-Namen.

In der obigen Beispiel-Bucket-Richtlinie ist *Bucket-Name* ein Platzhalter für den Bucket-Namen. Um diese Bucket-Richtlinie mit Ihrem eigenen Bucket zu verwenden, müssen Sie diesen Namen so aktualisieren, dass er mit Ihrem Bucket übereinstimmt.

6. Wählen Sie Save Changes (Änderungen speichern).

   Es wird eine Meldung angezeigt, die darauf hinweist, dass die Bucket-Richtlinie erfolgreich hinzugefügt wurde.

   Wenn die Fehlermeldung `Policy has invalid resource` angezeigt wird, bestätigen Sie, dass der Bucket-Name in der Bucket-Richtlinie mit Ihrem Bucket-Namen übereinstimmt. Informationen zum Hinzufügen einer Bucket-Richtlinie finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu?

   Wenn Sie eine Fehlermeldung erhalten und die Bucket-Richtlinie nicht speichern können, überprüfen Sie Ihr Konto und die Bucket-Einstellungen für Block Public Access, um zu bestätigen, dass Sie den öffentlichen Zugriff auf den Bucket zulassen.

Im nächsten Schritt können Sie die Endpunkte Ihrer Website herausfinden und Ihren Domänen-Endpunkt testen.

Schritt 10: Testen Ihres Domänen-Endpunkts

Nachdem Sie den Stammdomänen-Bucket zum Hosten einer öffentlichen Website konfiguriert haben, können Sie Ihren Endpunkt testen. Weitere Informationen finden Sie unter Website-Endpunkte. Sie können nur den Endpunkt für Ihren Domänen-Bucket testen, da Ihr Subdomänen-Bucket für die Website-Umleitung und nicht für das statische Website-Hosting eingerichtet ist.

> **Note**
>
> Amazon S3 unterstützt keinen HTTPS-Zugriff auf die Website. Wenn Sie HTTPS verwenden möchten, können Sie Amazon verwenden, CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird.
> Weitere Informationen finden Sie unter Wie verwende ich , CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird? und Erzwingen von HTTPS für die Kommunikation zwischen Viewern und CloudFront.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Properties (Eigenschaften).
3. Wählen Sie unten auf der Seite unter Static website hosting (Hosting statischer Websites) Ihren Bucket-Website-Endpunkt.

   Ihr Indexdokument wird in einem separaten Browserfenster geöffnet.

Im nächsten Schritt verwenden Sie Amazon Route 53, um den Kunden zu ermöglichen, Ihre Site über Ihre beiden benutzerdefinierten URLs aufzurufen.

Schritt 11: Hinzufügen von Aliasdatensätzen für Ihre Domäne und Subdomäne

In diesem Schritt erstellen Sie die Aliasdatensätze, die Sie der gehosteten Zone für Ihre Domänenzuordnungen `example.com` und `www.example.com` hinzufügen. Statt IP-Adressen verwenden die Alias-Datensätze Amazon S3 Website-Endpunkte. Amazon Route 53 verwaltet ein Mapping zwischen den Alias-Datensätzen und den IP-Adressen, wo sich die Amazon-S3-Buckets befinden. Sie erstellen zwei Alias-Datensätze: einen für Ihre Stamm-Domäne und einen für Ihre Subdomäne.

Hinzufügen eines Aliasdatensatzes für Ihre Stamm-Domäne und Subdomäne

So fügen Sie Ihrer Stamm-Domäne einen Aliasdatensatz hinzu (**`example.com`**)

1. Öffnen Sie die Route 53-Konsole unter https://console.aws.amazon.com/route53/.

   > **Note**
   >
   > Wenn Sie Route 53 noch nicht verwenden, lesen Sie Schritt 1: Registrieren einer Domäne im Amazon-Route 53-Entwicklerhandbuch. Nachdem Sie Ihre Einrichtung abgeschlossen haben, können Sie mit den Anweisungen fortfahren.

2. Wählen Sie Hosted Zones (Gehostete Zonen) aus.
3. Wählen Sie in der Liste der gehosteten Zonen den Namen der gehosteten Zone aus, der dem Domänennamen entspricht.
4. Wählen Sie Create Record Set (Datensatz erstellen).
5. Wählen Sie Switch to wizard (Zu Assistent wechseln) aus.

> **Note**
>
> Wenn Sie Quick Create zum Erstellen Ihrer Aliaseinträge verwenden möchten, lesen Sie Konfigurieren von Route 53 zum Weiterleiten von Datenverkehr an einen S3-Bucket.

6. Wählen Sie Simple Routing (Einfaches Routing), und wählen Sie Next (Weiter).
7. Wählen Sie Define simple record (Einfachen Datensatz definieren).
8. Akzeptieren Sie unter Record name (Datensatzname) den Standardwert, bei dem es sich um den Namen Ihrer gehosteten Zone und Ihrer Domäne handelt.
9. Wählen Sie unter Value/Route traffic to (Wert/Datenverkehr weiterleiten zu) die Option Alias to S3 website endpoint (Alias zu S3-Website-Endpunkt) aus.
10. Wählen Sie die Region aus.
11. Wählen Sie den S3-Bucket.

    Der Bucket-Name sollte mit dem Namen übereinstimmen, der im Feld Name angezeigt wird. In der Liste Choose S3 Bucket (S3-Bucket auswählen) wird der Bucket-Name mit dem Amazon-S3-Website-Endpunkt für die Region angezeigt, in der der Bucket erstellt wurde, zum Beispiel `s3-website-us-west-1.amazonaws.com (example.com)`.

    Choose S3 bucket (S3-Bucket auswählen) listet einen Bucket auf, wenn:

    - Sie den Bucket als statische Website konfiguriert haben.
    - Der Name des Buckets mit dem Namen des Datensatzes übereinstimmt, den Sie anlegen.
    - Das aktuelle hat den Bucket AWS-Konto erstellt.

    Wenn Ihr Bucket nicht in der Auflistung Choose S3 bucket (S3-Bucket auswählen) angezeigt wird, geben Sie den Amazon-S3-Website-Endpunkt für die Region ein, in der der Bucket erstellt wurde, z. B. **`s3-website-us-west-2.amazonaws.com`**. Eine vollständige Liste der Amazon-S3-Website-Endpunkte finden Sie unter Amazon-S3-Website-Endpunkte. Weitere Informationen über das Alias-Target finden Sie unter Wert/Datenverkehr weiterleiten zu im Amazon-Route 53-Entwicklerhandbuch.
12. Wählen Sie unter Datensatztyp A – Leitet den Datenverkehr an eine IPv4-Adresse und einige AWS Ressourcen weiter.
13. Wählen Sie unter Evaluate target health (Zielzustand bewerten) die Option No (Nein).
14. Wählen Sie Define simple record (Einfachen Datensatz definieren).

So fügen Sie Ihrer Subdomäne () einen Alias-Datensatz hi (**`www.example.com`**)

1. Wählen Sie unter Configure records (Datensätze konfigurieren) die Option Define simple record (Einfachen Datensatz definieren) aus
2. Geben Sie unter Record name (Datensatzname) für Ihre Subdomäne `www` ein.
3. Wählen Sie unter Value/Route traffic to (Wert/Datenverkehr weiterleiten zu) die Option Alias to S3 website endpoint (Alias zu S3-Website-Endpunkt) aus.
4. Wählen Sie die Region aus.
5. Wählen Sie den S3-Bucket, zum Beispiel, `s3-website-us-west-2.amazonaws.com (www.example.com)`.

   Wenn Ihr Bucket nicht in der Auflistung Choose S3 bucket (S3-Bucket auswählen) angezeigt wird, geben Sie den Amazon-S3-Website-Endpunkt für die Region ein, in der der Bucket erstellt wurde, z. B. **`s3-website-us-west-2.amazonaws.com`**. Eine vollständige Liste der Amazon-S3-Website-Endpunkte finden Sie unter Amazon-S3-Website-Endpunkte. Weitere Informationen über das Alias-Target finden Sie unter Wert/Datenverkehr weiterleiten zu im Amazon-Route 53-Entwicklerhandbuch.
6. Wählen Sie unter Datensatztyp A – Leitet den Datenverkehr an eine IPv4-Adresse und einige AWS Ressourcen weiter.
7. Wählen Sie unter Evaluate target health (Zielzustand bewerten) die Option No (Nein).
8. Wählen Sie Define simple record (Einfachen Datensatz definieren).
9. Klicken Sie auf der Seite Configure records (Datensätze konfigurieren) auf Create records (Datensätze erstellen).

> **Note**
>
> Änderungen werden im Allgemeinen innerhalb von 60 Sekunden an alle Route 53-Server übertragen. Wenn die Übertragung abgeschlossen ist, können Sie den Datenverkehr an den Amazon-S3-Bucket leiten, indem Sie den Namen des Alias-Datensatzes angeben, den Sie in diesem Verfahren erstellt haben.

Hinzufügen eines Alias-Datensatzes für Ihre Stamm-Domäne und Subdomäne (alte Route 53-Konsole)

So fügen Sie Ihrer Stamm-Domäne einen Aliasdatensatz hinzu (**`example.com`**)

Die Route 53-Konsole wurde überarbeitet. In der Route 53-Konsole können Sie vorübergehend die alte Konsole verwenden. Wenn Sie sich für die Arbeit mit der alten Route 53-Konsole entscheiden, gehen Sie wie folgt vor.

1. Öffnen Sie die Route 53-Konsole unter https://console.aws.amazon.com/route53/.

   > **Note**
   >
   > Wenn Sie Route 53 noch nicht verwenden, lesen Sie Schritt 1: Registrieren einer Domäne im Amazon-Route 53-Entwicklerhandbuch. Nachdem Sie Ihre Einrichtung abgeschlossen haben, können Sie mit den Anweisungen fortfahren.

2. Wählen Sie Hosted Zones (Gehostete Zonen) aus.
3. Wählen Sie in der Liste der gehosteten Zonen den Namen der gehosteten Zone aus, der dem Domänennamen entspricht.
4. Wählen Sie Create Record Set (Datensatz erstellen).
5. Geben Sie die folgenden Werte an:

   Name

   Akzeptieren Sie den Standardwert, bei dem es sich um den Namen Ihrer gehosteten Zone und Ihrer Domäne handelt.

   Für die Stamm-Domäne müssen Sie keine zusätzlichen Informationen in das Feld Name eingeben.

   Typ

   Wählen Sie A – IPv4 address (IPv4-Adresse).

   Alias

   Wählen Sie Yes (Ja).

Alias-Ziel

Wählen Sie im Bereich S3 website endpoints (S3-Website-Endpunkte) der Liste Ihren Bucket-Namen aus.

Der Bucket-Name sollte mit dem Namen übereinstimmen, der im Feld Name angezeigt wird. In der Auflistung Alias Target (Alias-Ziel) folgt dem Bucket-Namen beispielsweise der Amazon-S3-Website-Endpunkt für die Region, in der der Bucket erstellt wurde, z. B. `example.com (s3-website-us-west-2.amazonaws.com)`. Alias Target (Alias-Ziel) listet einen Bucket auf, wenn:

- Sie den Bucket als statische Website konfiguriert haben.
- Der Name des Buckets mit dem Namen des Datensatzes übereinstimmt, den Sie anlegen.
- Das aktuelle hat den Bucket AWS-Konto erstellt.

Wenn Ihr Bucket nicht in der Auflistung Alias Target (Alias-Ziel) angezeigt wird, geben Sie den Amazon-S3-Website-Endpunkt für die Region ein, in der der Bucket erstellt wurde, z. B. `s3-website-us-west-2`. Eine vollständige Liste der Amazon-S3-Website-Endpunkte finden Sie unter Amazon-S3-Website-Endpunkte. Weitere Informationen über das Alias-Target finden Sie unter Value/Route Traffic to (Wert/Datenverkehr weiterleiten zu) im Amazon-Route 53-Entwicklerhandbuch.

Routing-Richtlinie

Übernehmen Sie den Standardwert Simple.

Evaluate Target Health

Übernehmen Sie den Standardwert No.

6. Wählen Sie Create (Erstellen) aus.

So fügen Sie Ihrer Subdomäne () einen Alias-Datensatz hi (**`www.example.com`**)

1. Wählen Sie in der gehosteten Zone für Ihre Stamm-Domäne (`example.com`) die Option Create record set (Datensatz erstellen) aus.
2. Geben Sie die folgenden Werte an:

   Name

   Geben Sie als Subdomäne „www“ in das Feld ein.

Typ

Wählen Sie A – IPv4 address (IPv4-Adresse).

Alias

Wählen Sie Yes (Ja).

Alias-Ziel

Wählen Sie im Abschnitt S3 website endpoints (S3-Website-Endpunkte) der Liste denselben Bucket-Namen aus, der im Feld Name angezeigt wird (z. B. `www.example.com (s3-website-us-west-2.amazonaws.com)`).

Routing-Richtlinie

Übernehmen Sie den Standardwert Simple.

Evaluate Target Health

Übernehmen Sie den Standardwert No.

3. Wählen Sie Create aus.

> **Note**
>
> Änderungen werden im Allgemeinen innerhalb von 60 Sekunden an alle Route 53-Server übertragen. Wenn die Übertragung abgeschlossen ist, können Sie den Datenverkehr an den Amazon-S3-Bucket leiten, indem Sie den Namen des Alias-Datensatzes angeben, den Sie in diesem Verfahren erstellt haben.

Schritt 12: Testen der Website

Bestätigen Sie, dass die Website und die Umleitung korrekt funktionieren. Geben Sie Ihre URLs in Ihren Browser ein. In diesem Beispiel können Sie folgende URLs testen:

- Domain (Domäne) (`http://example.com`) – Zeigt das Indexdokument im `example.com`-Bucket an.
- Subdomain (Subdomäne) (`http://www.example.com`) – Leitet Ihre Anforderung an `http://example.com` weiter. Sie sehen das Indexdokument im `example.com`-Bucket.

Wenn Ihre Website oder Weiterleitungslinks nicht funktionieren, können Sie Folgendes versuchen:

- Clear cache (Cache löschen) – Löschen Sie den Cache Ihres Webbrowsers.
- Check name servers (Nameserver überprüfen) – Wenn Ihre Webseite und Weiterleitungslinks nach dem Löschen des Caches nicht funktionieren, können Sie die Namenserver für Ihre Domäne und die Namenserver für Ihre gehostete Zone vergleichen. Wenn die Namenserver nicht übereinstimmen, müssen Sie möglicherweise Ihre Domänennamenserver so aktualisieren, dass sie mit den unter Ihrer gehosteten Zone aufgelisteten übereinstimmen. Weitere Informationen finden Sie unter Hinzufügen oder Ändern von Namenservern und Kleben von Datensätzen für eine Domäne.

Nachdem Sie Ihre Stamm- und Subdomäne erfolgreich getestet haben, können Sie eine Amazon CloudFront-Verteilung einrichten, um die Leistung Ihrer Website zu verbessern und Protokolle bereitzustellen, mit denen Sie den Website-Datenverkehr überprüfen können. Weitere Informationen finden Sie unter Beschleunigen Ihrer Website mit Amazon CloudFront.

Beschleunigen Ihrer Website mit Amazon CloudFront

Sie können Amazon CloudFront verwenden, um die Leistung Ihrer Amazon S3-Website zu verbessern. CloudFront erstellt Ihre Website-Dateien (wie HTML, Bilder und Videos), die von Rechenzentren auf der ganzen Welt (als Edge-Standorte bezeichnet) verfügbar sind. Wenn ein Besucher eine Datei von Ihrer Website anfordert, leitet CloudFront die Anforderung automatisch zu einer Kopie der Datei am nächstgelegenen Edge-Standort um. Das führt zu schnelleren Download-Zeiten, als wenn der Besucher den Inhalt von einem weitere entfernteren Rechenzentrum angefordert hätte.

CloudFront speichert Inhalte an Edge-Standorten für einen von Ihnen angegebenen Zeitraum zwischen. Wenn ein Besucher Inhalte anfordert, die länger als das Ablaufdatum zwischengespeichert wurden, CloudFront überprüft den Ursprungsserver, um festzustellen, ob eine neuere Version des Inhalts verfügbar ist. Wenn eine neuere Version verfügbar ist, CloudFront kopiert die neue Version an den Edge-Standort. Änderungen an den ursprünglichen Inhalten werden zu den Edge-Standorten repliziert, indem die Besucher die Inhalte anfordern.

Verwenden von CloudFront ohne Route 53

Das Tutorial auf dieser Seite verwendet Route 53, um auf Ihre CloudFront Verteilung zu verweisen. Wenn Sie jedoch Inhalte bereitstellen möchten, die in einem Amazon S3-Bucket mit CloudFront ohne Verwendung von Route 53 gehostet werden, finden Sie weitere Informationen unter Amazon

CloudFront-Tutorials: Einrichten einer dynamischen Inhaltsverteilung für Amazon S3. Wenn Sie Inhalte bereitstellen, die in einem Amazon S3-Bucket mit gehostet werden CloudFront, können Sie einen beliebigen Bucket-Namen verwenden, und sowohl HTTP als auch HTTPS werden unterstützt.

Automatisieren der Einrichtung mit einer - AWS CloudFormation Vorlage

Weitere Informationen zur Verwendung einer - AWS CloudFormation Vorlage zum Konfigurieren einer sicheren statischen Website, die eine CloudFront Verteilung für Ihre Website erstellt, finden Sie unter Erste Schritte mit einer sicheren statischen Website im Amazon- CloudFront Entwicklerhandbuch.

Themen

Schritt 1: Erstellen einer CloudFront Verteilung

Zunächst erstellen Sie eine CloudFront Verteilung. Damit steht Ihre Website in weltweit angesiedelten Rechenzentren zur Verfügung.

So erstellen Sie eine Verteilung mit einem Amazon S3-Ursprung

1. Öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Wählen Sie Create Distribution (Distribution erstellen).
3. Geben Sie auf der Seite Create Distribution (Verteilung erstellen) im Abschnitt Origin Settings (Ursprungseinstellungen) unter Origin Domain Name (Ursprungsdomänenname) den Amazon S3-Website-Endpunkt für Ihren Bucket ein – z. B. **`example.com.s3-website.us-west-1.amazonaws.com`**.

   CloudFront füllt die Ursprungs-ID für Sie aus.
4. Behalten Sie für Default Cache Behavior Settings (Standardeinstellungen für das Zwischenspeicherverhalten) die Standardwerte bei.

   Mit den Standardeinstellungen für Viewer Protocol Policy (Viewer-Protokollrichtlinie) können Sie HTTPS für Ihre statische Website verwenden. Weitere Informationen zu diesen Konfigurationsoptionen finden Sie unter Werte, die Sie beim Erstellen oder Aktualisieren einer Web-Verteilung angeben im Amazon- CloudFront Entwicklerhandbuch.

5. Gehen Sie unter Distribution Settings (Verteilungseinstellungen) wie folgt vor:

   a. Lassen Sie für die Option Price Class (Preisklasse) die Einstellung Use All Edge Locations (Best Performance) (Alle Edge-Standorte verwenden (Beste Leistung)) unverändert.

   b. Legen Sie beispielsweise Alternate Domain Names (CNAMEs) (Alternative Domänennamen (CNAMEs)) für die Stammdomäne und die `www`-Unterdomäne fest. In diesem Tutorial sind dies `example.com` und `www.example.com`.

      > **Important**
      >
      > Bevor Sie diesen Schritt durchführen, achten Sie auf die Anforderungen für die Verwendung alternativer Domänennamen, insbesondere hinsichtlich eines erforderlichen gültigen SSL/TLS-Zertifikats.

   c. Wählen Sie für SSL Certificate (SSL-Zertifikat) die Option Custom SSL Certificate (example.com) (Benutzerdefiniertes SSL-Zertifikat (beispiel.com)) und das benutzerdefinierte Zertifikat aus, das die Domänen- und Subdomänennamen abdeckt.

      Weitere Informationen finden Sie unter SSL-Zertifikat im Amazon- CloudFront Entwicklerhandbuch.

   d. Geben Sie in Default Root Object (Standard-Root-Objekt) den Namen Ihres Indexdokuments ein, z. B. `index.html`.

      Wenn die URL für den Zugriff auf die Verteilung keinen Dateinamen enthält, gibt die CloudFront Verteilung das Indexdokument zurück. Das Default Root Object (Standard-Root-Objekt) sollte genau mit dem Namen des Indexdokuments für Ihre statische Website übereinstimmen. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments.

   e. Wählen Sie für Logging (Protokollierung) die Option On (Ein).

      > **Important**
      >
      > Wenn Sie eine Verteilung erstellen oder aktualisieren und die CloudFront Protokollierung aktivieren, CloudFront aktualisiert die Bucket-Zugriffssteuerungsliste (ACL), um dem `awslogsdelivery` Konto `FULL_CONTROL` Berechtigungen zum Schreiben von Protokollen in Ihren Bucket zu erteilen. Weitere Informationen finden Sie unter Erforderliche Berechtigungen zum Konfigurieren der Standardprotokollierung und zum Zugriff auf Ihre Protokolldateien im Amazon-

CloudFront Entwicklerhandbuch. Wenn der Bucket, der die Protokolle speichert, die Einstellung „Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, um ACLs zu deaktivieren, CloudFront kann keine Protokolle in den Bucket schreiben. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

f. Wählen Sie als Bucket for Logs (Bucket für Protokolle) den Bucket zur Protokollierung aus, den Sie erstellt haben.

   Weitere Informationen zum Konfigurieren eines Protokoll-Buckets finden Sie unter (Optional) Protokollieren des Webdatenverkehrs.

g. Wenn Sie die Protokolle speichern möchten, die durch den Datenverkehr zur CloudFront - Verteilung generiert werden, geben Sie in Log Prefix (Protokollpräfix) den Ordnernamen ein.

h. Behalten Sie für alle übrigen Einstellungen die Standardwerte bei.

6. Wählen Sie Create Distribution.
7. Um den aktuellen Status der Verteilung anzuzeigen, suchen Sie die Verteilung in der Konsole, und prüfen Sie die Spalte Status.

   Der Status `InProgress` gibt an, dass die Verteilung noch nicht vollständig bereitgestellt ist.

   Wenn die Verteilung bereitgestellt wurde, können Sie Ihren Inhalt mit dem neuen CloudFront-Domain-Namen referenzieren.
8. Notieren Sie sich den Wert von Domain Name, der in der CloudFront Konsole angezeigt wird, z. B. `dj4p1rv6mvubz.cloudfront.net`.
9. Um zu überprüfen, ob Ihre CloudFront Verteilung funktioniert, geben Sie den Domänennamen der Verteilung in einen Webbrowser ein.

   Wenn Ihre Website sichtbar ist, funktioniert die CloudFront Verteilung. Wenn Ihre Website eine benutzerdefinierte Domäne hat, die bei Amazon Route 53 registriert ist, benötigen Sie den CloudFront Domänennamen, um den Datensatz im nächsten Schritt zu aktualisieren.

Schritt 2: Aktualisieren der Datensätze für Ihre Domäne und Unterdomäne

Nachdem Sie nun erfolgreich eine CloudFront Verteilung erstellt haben, aktualisieren Sie den Aliasdatensatz in Route 53 so, dass er auf die neue CloudFront Verteilung verweist.

So aktualisieren Sie den Aliasdatensatz so, dass er auf eine CloudFront Verteilung verweist

1. Öffnen Sie die Route 53-Konsole unter https://console.aws.amazon.com/route53/.
2. Wählen Sie in der linken Navigation Hosted Zones (Gehostete Zonen).
3. Wählen Sie auf der Seite Hosted Zones (Gehostete Zonen) die gehostete Zone aus, die Sie für Ihre Unterdomäne erstellt haben, beispielsweise `www.example.com`.
4. Wählen Sie unter Datensätzeden A-Eintrag aus, den Sie für Ihre Unterdomäne erstellt haben.
5. Wählen Sie unter Datensatzdetailsden Befehl Datensatz bearbeiten.
6. Wählen Sie unter Datenverkehr an weiterleiten die Option Alias zur CloudFront Verteilung aus.
7. Wählen Sie unter Verteilung auswählen die CloudFront Verteilung aus.
8. Wählen Sie Speichern.
9. Um den A-Datensatz für die Stammdomäne an die CloudFront Verteilung umzuleiten, wiederholen Sie dieses Verfahren für die Stammdomäne, z. B. `example.com`.

   Die Aktualisierung der Datensätze wird innerhalb von 2 bis 48 Stunden wirksam.
10. Um zu sehen, ob die neuen A-Datensätze wirksam sind, geben Sie in einem Webbrowser die URL Ihrer Unterdomäne ein, z. B. `http://www.example.com`.

    Wenn der Browser Sie nicht mehr zur Stammdomäne umleitet (z. B. `http://example.com`), sind die neuen A-Datensätze vorhanden. Wenn der neue A-Datensatz wirksam ist, wird der Datenverkehr, der vom neuen A-Datensatz an die CloudFront Verteilung weitergeleitet wird, nicht an die Stammdomäne umgeleitet. Besucher, die die Website mithilfe von `http://example.com` oder referenzieren, `http://www.example.com` werden an den nächstgelegenen CloudFront Edge-Standort umgeleitet, wo sie von schnelleren Download-Zeiten profitieren.

> **Tip**
>
> Umleitungseinstellungen können von Browsern zwischengespeichert werden. Wenn Sie annehmen, dass die neuen A-Datensatzeinstellungen wirksam sind, aber trotzdem eine Umleitung von `http://www.example.com` nach `http://example.com` feststellen, löschen Sie zum Testen den Verlauf und den Cache des Browsers. Schließen Sie den Browser und öffnen Sie ihn wieder oder verwenden Sie einen anderen Webbrowser, wenn Sie einen weiteren installiert haben.

(Optional) Schritt 3: Überprüfen der Protokolldateien

Die Zugriffsprotokolle teilen Ihnen mit, wie viele Menschen die Website besuchen. Sie enthalten auch wertvolle Geschäftsdaten, die Sie mithilfe anderer Services wie beispielsweise Amazon EMR analysieren können.

CloudFront -Protokolle werden in dem Bucket und Ordner gespeichert, den Sie auswählen, wenn Sie eine CloudFront Verteilung erstellen und logging. CloudFront writes-Protokolle in Ihren Protokoll-Bucket innerhalb von 24 Stunden ab dem Zeitpunkt der entsprechenden Anforderungen aktivieren.

Die Protokolldateien für Ihre Website anzeigen

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Protokoll-Bucket für Ihre Website aus.
3. Wählen Sie den Ordner CloudFront Protokolle aus.
4. Laden Sie die von geschriebenen `.gzip` Dateien herunter, CloudFront bevor Sie sie öffnen.

   Wenn Sie die Website nur zur Übung erstellt haben, können Sie die von Ihnen zugewiesenen Ressourcen löschen, damit keine weiteren Kosten für Sie anfallen. Lesen Sie dazu den Abschnitt Bereinigung Ihrer Beispielressourcen. Nach der Löschung der AWS -Ressourcen ist die Website nicht mehr verfügbar.

Bereinigung Ihrer Beispielressourcen

Wenn Sie die statische Website zur Übung erstellt haben, sollten Sie die zugewiesenen AWS -Ressourcen löschen, damit keine weiteren Kosten für Sie anfallen. Nach der Löschung der AWS -Ressourcen ist die Website nicht mehr verfügbar.

Aufgaben

- Schritt 1: Löschen der Amazon CloudFront-Verteilung
- Schritt 2: Löschen der von Route 53 gehosteten Zone
- Schritt 3: Deaktivieren der Protokollierung und Löschen Ihres S3-Buckets

Schritt 1: Löschen der Amazon CloudFront-Verteilung

Bevor Sie eine Amazon- CloudFront Verteilung löschen, müssen Sie sie deaktivieren. Eine deaktivierte Verteilung funktioniert nicht mehr und es fallen keine weiteren Kosten für sie an. Sie

können eine deaktivierte Verteilung jederzeit wieder aktivieren. Nachdem Sie eine deaktivierte Verteilung gelöscht haben, ist sie nicht länger verfügbar.

So deaktivieren und löschen Sie eine CloudFront Verteilung

1. Öffnen Sie die - CloudFront Konsole unter https://console.aws.amazon.com/cloudfront/v4/home.
2. Klicken Sie mit der rechten Maustaste auf die Verteilung, die Sie deaktivieren möchten, und anschließend auf Disable (Deaktivieren).
3. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Yes, Disable (Ja, deaktivieren).
4. Wählen Sie die deaktivierte Verteilung aus, und klicken Sie dann auf Delete (Löschen).
5. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Yes, Delete.

Schritt 2: Löschen der von Route 53 gehosteten Zone

Bevor Sie eine gehostete Zone löschen können, müssen Sie die zuvor erstellten Datensätze löschen. SOA-Einträge (Start of Authority, Autoritätsursprung) und NS-Einträge (Nameserver, Namenserver) müssen nicht von Ihnen entfernt werden, da dies automatisch beim Löschen der gehosteten Zone ausgeführt wird.

Sie löschen die Datensätze wie folgt:

1. Öffnen Sie die Route 53-Konsole unter https://console.aws.amazon.com/route53/.
2. Wählen Sie in der Liste der Domänennamen Ihren Domänennamenaus, und wählen Sie anschließend Go to Record Sets (Zu den Datensätzen).
3. Wählen Sie in der Liste der Datensatzsätze die A-Datensätze aus, die Sie erstellt haben.

   Der einzelne Datensatztyp wird in der Spalte Type (Typ) aufgeführt.
4. Wählen Sie Delete Record Set (Datensatz löschen).
5. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Confirm (Bestätigen).

So löschen Sie eine von Route 53 gehostete Zone:

1. Setzen Sie das vorherige Verfahren fort, indem Sie Back to Hosted Zones (Zurück zu gehosteten Zonen) auswählen.
2. Wählen Sie Ihren Domänennamen und dann Delete Hosted Zone (Gehostete Zone löschen).
3. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Confirm (Bestätigen).

Schritt 3: Deaktivieren der Protokollierung und Löschen Ihres S3-Buckets

Bevor Sie Ihren S3-Bucket löschen, stellen Sie sicher, dass die Protokollierung für den Bucket deaktiviert ist. Andernfalls schreibt AWS weiterhin Protokolle in Ihren Bucket, wenn Sie ihn löschen.

Deaktivieren Sie die Protokollierung für einen Bucket wie folgt:

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie unter Buckets (Buckets) den Namen Ihres Buckets aus. Wählen Sie anschließend Properties (Eigenschaften) aus.
3. Wählen Sie unter Properties (Eigenschaften) Logging (Protokollierung) aus.
4. Deaktivieren Sie das Kontrollkästchen Enabled (Aktiviert).
5. Wählen Sie Save (Speichern) aus.

Jetzt können Sie Ihren Bucket löschen. Weitere Informationen finden Sie unter Löschen eines Buckets.

Erstellen, Konfigurieren und Arbeiten mit Amazon S3-Buckets

Um Ihre Daten in Amazon S3 zu speichern, arbeiten Sie mit Ressourcen, die als Buckets und Objekte bezeichnet werden. Ein Bucket ist ein Container für Objekte. Ein Objekt ist eine Datei und alle Metadaten, die diese Datei beschreiben.

Um ein Objekt in Amazon S3 zu speichern, erstellen Sie einen Bucket und laden das Objekt dann in einen Bucket hoch. Wenn sich das Objekt im Bucket befindet, können Sie es öffnen, herunterladen und verschieben. Wenn Sie kein Objekt oder einen Bucket mehr benötigen, können Sie Ihre Ressourcen aufräumen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

> **Note**
>
> Mit Amazon S3 zahlen Sie nur für das, was Sie tatsächlich nutzen. Weitere Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon S3. Wenn Sie neuer Amazon-S3-Kunde sind, können Sie kostenlos mit Amazon S3 beginnen. Weitere Informationen finden Sie unter Kostenloses Kontingent für AWS.

Die Themen in diesem Abschnitt geben einen Überblick über die Arbeit mit Buckets in Amazon S3. Sie enthalten Informationen über das Benennen, Erstellen, Zugreifen und Löschen von Buckets. Weitere Informationen zum Anzeigen oder Auflisten von Objekten in einem Bucket finden Sie unter Organisieren, Auflisten und Arbeiten mit Ihren Objekten.

Themen

- Erstellen eines Buckets
- Anzeigen der Eigenschaften eines S3-Buckets
- Leeren eines Buckets
- Löschen eines Buckets
- Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets
- Arbeiten mit Mountpoint für Amazon S3
- Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration
- Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung
- Beschränkungen und Einschränkungen von Buckets

Bucket-Übersicht

Zum Hochladen Ihrer Daten (Fotos, Videos, Dokumente usw.) in Amazon S3 müssen Sie zunächst einen S3-Bucket in einer AWS-Regionen erstellen.

Ein Bucket ist ein Behälter für Objekte, die in Amazon S3 gespeichert werden. Sie können beliebig viele Objekte in einem Bucket speichern und bis zu 100 Buckets in Ihrem Konto haben. Um eine Erhöhung anzufordern, rufen Sie die Service-Quotas-Konsole auf.

Jedes Objekt ist in einem Bucket enthalten. Wenn beispielsweise ein Objekt mit dem Namen `photos/puppy.jpg` im Bucket `DOC-EXAMPLE-BUCKET` in der Region USA West (Oregon) gespeichert ist, ist es über die URL `https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg` adressierbar. Weitere Informationen siehe Zugriff auf einen Bucket.

In Bezug auf die Implementierung sind Buckets und Objekte AWS Ressourcen, und Amazon S3 stellt APIs für deren Verwaltung bereit. Beispielsweise können Sie mit der Amazon-S3-API einen Bucket erstellen und Objekte hochladen. Sie können diese Vorgänge auch in der Amazon-S3-Konsole durchführen. Die Konsole verwendet die Amazon-S3-APIs, um Anfragen an Amazon S3 zu senden.

In diesem Abschnitt wird beschrieben, wie Buckets verwendet werden. Weitere Informationen zur Arbeit mit Objekten finden Sie unter Übersicht über Amazon-S3-Objekte.

Amazon S3 unterstützt globale Buckets, was bedeutet, dass jeder Bucket-Name in allen AWS-Konten innerhalb AWS-Regionen einer Partition eindeutig sein muss. Eine Partition ist eine Gruppierung von Regionen. AWS verfügt derzeit über drei Partitionen: `aws` (Standardregionen), `aws-cn` (China-Regionen) und `aws-us-gov` (AWS GovCloud (US)).

Nachdem ein Bucket erstellt wurde, kann der Name dieses Buckets erst von einem anderen AWS-Konto in derselben Partition verwendet werden, wenn der Bucket gelöscht wurde. Verlassen Sie sich nicht auf eine spezifische Benennungskonvention für Buckets für Verfügbarkeits- oder Sicherheitsprüfungszwecke. Namensrichtlinien für Buckets finden Sie unter Regeln für die Benennung von Buckets.

Amazon S3 erstellt Buckets in der von Ihnen angegebenen Region. Um die Latenz zu reduzieren, Kosten zu minimieren oder gesetzliche Anforderungen zu erfüllen, wählen Sie eine aus, AWS-Region die geografisch in Ihrer Nähe liegt. Wenn Sie beispielsweise in Europa ansässig sind, könnte es vorteilhaft sein, Buckets in den Regionen EU (Irland) oder EU (Frankfurt) zu erstellen. Eine Liste der Amazon-S3-Regionen finden Sie unter Regions and Endpoints (Regionen und Endpunkte) in der Allgemeinen AWS -Referenz.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

> **Note**
>
> Objekte, die zu einem Bucket gehören, den Sie in einer bestimmten erstellen, verlassen diese Region AWS-Region niemals, es sei denn, Sie übertragen sie explizit in eine andere Region. Beispielsweise verlassen in der Region EU (Irland) gespeicherte Objekte diese Region nie.

Themen

Berechtigungen

Sie können Ihre - Root-Benutzer des AWS-Kontos Anmeldeinformationen verwenden, um einen Bucket zu erstellen und einen anderen Amazon S3-Vorgang auszuführen. Wir empfehlen jedoch, nicht die Root-Benutzer-Anmeldeinformationen Ihres AWS-Konto für Anfragen zu verwenden, z. B.

zum Erstellen eines Buckets. Erstellen Sie stattdessen einen AWS Identity and Access Management (IAM)-Benutzer und gewähren Sie diesem Benutzer Vollzugriff (Benutzer haben standardmäßig keine Berechtigungen).

Diese Benutzer werden als Administratorenbezeichnet. Sie können die Anmeldeinformationen des Administratorbenutzers anstelle der Root-Benutzer-Anmeldeinformationen Ihres -Kontos verwenden, um mit zu interagieren AWS und Aufgaben auszuführen, z. B. um einen Bucket zu erstellen, Benutzer zu erstellen und ihnen Berechtigungen zu erteilen.

Weitere Informationen finden Sie unter Root-Benutzer des AWS-Kontos -Anmeldeinformationen und IAM-Benutzer-Anmeldeinformationen in der Allgemeinen AWS -Referenz und unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Das AWS-Konto , das eine Ressource erstellt, besitzt diese Ressource. Wenn Sie beispielsweise einen IAM-Benutzer in Ihrem erstellen AWS-Konto und dem Benutzer die Berechtigung zum Erstellen eines Buckets erteilen, kann der Benutzer einen Bucket erstellen. Der Benutzer besitzt den Bucket jedoch nicht. Der AWS-Konto , zu dem der Benutzer gehört, besitzt den Bucket. Der Benutzer benötigt eine zusätzliche Berechtigung vom Ressourceneigentümer, um andere Bucket-Operation auszuführen. Weitere Informationen zum Verwalten von Berechtigungen für Ihre Amazon-S3-Ressourcen finden Sie unter Identity and Access Management in Amazon S3.

Verwalten des öffentlichen Zugriffs auf Buckets

Öffentlicher Zugriff auf Buckets und Objekte wird über Bucket-Richtlinien, Zugriffssteuerungslisten (ACLs) oder beides gewährt. Um Sie bei der Verwaltung des öffentlichen Zugriffs auf Amazon-S3-Ressourcen zu unterstützen, bietet Amazon S3 Einstellungen für das Sperren des öffentlichen Zugriffs. Amazon S3 Block Public Access-Einstellungen können ACLs und Bucket-Richtlinien überschreiben, damit Sie einheitliche Einschränkungen des öffentlichen Zugriffs auf diese Ressourcen durchsetzen können. Sie können Block Public Access-Einstellungen auf einzelne Buckets oder auf alle Buckets in Ihrem Konto anwenden.

Um zu gewährleisten, dass bei allen Amazon-S3-Buckets und -Objekten der öffentliche Zugriff blockiert ist, sind alle vier Einstellungen für Block Public Access standardmäßig aktiviert, wenn Sie einen neuen Bucket erstellen. Wir empfehlen, alle vier Einstellungen für Block Public Access auch für Ihr Konto zu aktivieren. Diese Einstellungen blockieren den öffentlichen Zugriff für alle aktuellen und künftigen Buckets.

Bevor Sie diese Einstellungen anwenden, verifizieren Sie, dass Ihre Anwendungen ohne öffentlichen Zugriff korrekt funktionieren. Wenn ein bestimmtes Maß an öffentlichem Zugriff auf Ihre Buckets

oder Objekte nötig ist, z. B. zum Hosten einer statischen Website, wie unter Hosten einer statischen Website mit Amazon S3 beschrieben, können Sie die einzelnen Einstellungen an Ihre Speicheranwendungsfälle anpassen. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Wir empfehlen jedoch dringend, Block Public Access aktiviert zu lassen. Wenn Sie alle vier Block Public Access-Einstellungen aktiviert lassen und eine statische Website hosten möchten, können Sie die Amazon- CloudFront Ursprungszugriffssteuerung (OAC) verwenden. Amazon CloudFront bietet die Funktionen, die zum Einrichten einer sicheren statischen Website erforderlich sind. Statische Amazon-S3-Websites unterstützen nur HTTP-Endpunkte. Amazon CloudFront verwendet den dauerhaften Speicher von Amazon S3 und bietet gleichzeitig zusätzliche Sicherheitsheader wie HTTPS. HTTPS erhöht die Sicherheit, indem eine normale HTTP-Anforderung verschlüsselt und vor gängigen Cyberangriffen geschützt wird.

Weitere Informationen finden Sie unter Erste Schritte mit einer sicheren statischen Website im Amazon- CloudFront Entwicklerhandbuch.

> **Note**
>
> Wenn beim Auflisten Ihrer Buckets mit ihren Einstellungen für den öffentlichen Zugriff die Fehlermeldung `Error` angezeigt wird, verfügen Sie möglicherweise nicht über die erforderlichen Berechtigungen. Stellen Sie sicher, dass Sie Ihrer Benutzer- oder Rollenrichtlinie die folgenden Berechtigungen hinzugefügt haben:
>
> ```
> s3:GetAccountPublicAccessBlock
> s3:GetBucketPublicAccessBlock
> s3:GetBucketPolicyStatus
> s3:GetBucketLocation
> s3:GetBucketAcl
> s3:ListAccessPoints
> s3:ListAllMyBuckets
> ```
>
> In einigen seltenen Fällen können Anfragen auch aufgrund des Ausfalls einer AWS-Region fehlschlagen.

Optionen für die Bucket-Konfiguration

Amazon S3 unterstützt Vorgänge Optionen für die Konfiguration Ihres Buckets. Sie können Ihren Bucket beispielsweise für ein Website-Hosting konfigurieren, eine Konfiguration zur Verwaltung des Lebenszyklus von Objekten im Bucket hinzufügen und den Bucket so konfigurieren, dass alle Zugriffe protokolliert werden. Amazon S3 unterstützt Subressourcen, um die Bucket-Konfigurationsinformationen zu speichern und zu verwalten. Sie können diese Subressourcen mit der Amazon-S3-API erstellen und verwalten. Sie können jedoch auch die -Konsole oder die - AWS SDKs verwenden.

> **Note**
>
> Außerdem gibt es Konfigurationen auf Objektebene. Beispielsweise können Sie Berechtigungen auf Objektebene konfigurieren, indem Sie eine für dieses Objekt spezifische Zugriffskontrollliste (ACL) konfigurieren.

Man bezeichnet sie als Subressourcen, weil sie im Kontext eines spezifischen Buckets oder Objekts existieren. Die folgende Tabelle listet Subressourcen auf, die ihnen ermöglichen, für den Bucket spezifische Konfigurationen zu verwalten.

Subressource	Beschreibung
cors (ursprung sübergreifende gemeinsame Nutzung von Ressourcen)	Sie können Ihren Bucket so konfigurieren, dass er ursprungsübergreifende Anforderungen zulässt. Weitere Informationen finden Sie unter Cross-Origin Resource Sharing (CORS) verwenden.
event notification	Sie können Ihrem Bucket gestatten, Benachrichtigungen über bestimmte Bucket-Ereignisse zu senden. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.
Lebenszyklus	Sie können Lebenszyklusregeln für Objekte in Ihrem Bucket mit definiertem Lebenszyklus definieren. Sie können z. B. eine Regel so festlegen, dass

Subressource	Beschreibung
	Objekte ein Jahr nach dem Erstellungsdatum archiviert werden, oder dass ein Objekt 10 Jahre nach dem Erstellen gelöscht wird. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.
location	Wenn Sie einen Bucket erstellen, geben Sie die an, AWS-Region in der Amazon S3 den Bucket erstellen soll. Amazon S3 speichert diese Informationen in der Subressource location und stellt Ihnen eine API bereit, mit der Sie diese Informationen abrufen können.
logging	Protokollierung ermöglicht Ihnen, Zugriffsanforderungen für Ihren Bucket nachzuverfolgen. Jeder Zugriffsprotokolldatensatz enthält Details über eine Zugriffsanforderung, z. B. Auftraggeber, Bucket-Name, Anforderungszeit, Anforderungsaktion, Antwortstatus und Fehlercode, falls vorhanden. Die Zugriffsprotokollinformationen können für Sicherheits- und Zugriffsüberprüfungen nützlich sein. Außerdem erfahren Sie damit mehr über Ihren Kundenstamm und erhalten einen Überblick über Ihre Amazon-S3-Rechnung. Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.
Objektsperre	Zur Verwendung der S3-Objektsperre muss diese für einen Bucket aktiviert werden. Sie können optional auch einen Standardaufbewahrungsmodus und -zeitraum konfigurieren, der für neue Objekte gilt, die in den Bucket platziert werden. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.
policy und ACL (Access Control List, Zugriffskontrollliste)	Standardmäßig sind alle Ihre Ressourcen (wie Buckets und Objekte) privat. Amazon S3 unterstützt die Optionen Bucket-Richtlinie und Zugriffskontrollliste (ACL) für Sie, um Berechtigungen auf Bucket-Ebene zu erteilen und zu verwalten. Amazon S3 speichert die Berechtigungsinformationen in den Subressourcen policy und acl. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3.

Subressource	Beschreibung
Replikation	Die Replikation ist ein automatisches, asynchrones Kopieren von Objekten über Buckets hinweg in derselben oder in verschiedenen AWS-Regionen. Weitere Informationen finden Sie unter Replizieren von Objekten.
requestPayment	Standardmäßig zahlt das , AWS-Konto das den Bucket erstellt (der Bucket-Eigentümer), für Downloads aus dem Bucket. Mit dieser Subressource kann der Bucket-Eigentümer angeben, dass die Person, die den Download anfordert, die Gebühren für den Download trägt. Amazon S3 stellt eine API für Sie bereit, mit der Sie diese Subressource verwalten können. Weitere Informationen finden Sie unter Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung.
Markieren	Sie können Ihrem Bucket Kostenzuordnungs-Tags hinzufügen, um Ihre AWS Kosten zu kategorisieren und zu verfolgen. Amazon S3 unterstützt die tagging-Subressource, um Markierungen für einen Bucket zu speichern und zu verwalten. Mithilfe von Tags, die Sie auf Ihren Bucket anwenden, AWS generiert einen Kostenzuordnungsbericht mit Nutzung und Kosten, aggregier t nach Ihren Tags. Weitere Informationen finden Sie unter Fakturierungs- und Nutzungsberichte für Amazon S3.
transfer accelerat ion	Transfer Acceleration ermöglicht schnelle, einfache und sichere Übertragu ngen von Dateien zwischen Ihrem Client und einem S3-Bucket über große Entfernungen. Transfer Acceleration nutzt die global verteilten Edge-Stan dorte von Amazon CloudFront. Weitere Informationen finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.

Subressource	Beschreibung
Versioning	Versioning hilft Ihnen bei einer Wiederherstellung nach einem versehentlichen Überschreiben und Löschen. Wir empfehlen das Versioning als bewährte Methode, um zu verhindern, dass Objekte versehentlich gelöscht oder überschrieben werden. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.
Website	Sie können Ihren Bucket für ein Hosting statischer Websites konfigurieren. Amazon S3 speichert diese Konfiguration, indem es eine website-Subressource erstellt. Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.

Regeln für die Benennung von Buckets

Die folgenden Regeln gelten für die Benennung von Buckets für allgemeine Zwecke und von Verzeichnis-Buckets in Amazon S3:

Themen

- Regeln für die Benennung von Buckets für allgemeine Zwecke
- Regeln für die Benennung von Verzeichnis-Buckets

Regeln für die Benennung von Buckets für allgemeine Zwecke

Die folgenden Regeln gelten für die Benennung von Buckets für allgemeine Zwecke.

- Bucket-Namen müssen zwischen (min.) 3 und (max.) 63 Zeichen lang sein.
- Bucket-Namen können nur aus Kleinbuchstaben, Zahlen, Punkten (.) und Bindestrichen (-) bestehen.
- Bucket-Namen müssen mit einem Buchstaben oder einer Zahl beginnen und enden.
- Bucketnamen dürfen keine aufeinander folgenden Punkte (..) enthalten.

- Bucket-Namen dürfen nicht als IP-Adresse formatiert sein (zum Beispiel 192.168.5.4).
- Der Bucket-Name darf nicht mit dem Präfix `xn--` beginnen.
- Bucket-Namen dürfen nicht mit dem Präfix `sthree-` und dem Präfix `sthree-configurator` beginnen.
- Bucket-Namen dürfen nicht mit dem Suffix `-s3alias` enden. Dieses Suffix ist für Zugriffspunkt-Aliasnamen reserviert. Weitere Informationen finden Sie unter Verwenden eines Alias im Bucket-Stil für Ihren S3-Bucket-Zugriffspunkt.
- Bucket-Namen dürfen nicht mit dem Suffix `--ol-s3` enden. Dieses Suffix ist für Objekt-Lambda-Zugriffspunkt-Aliasnamen reserviert. Weitere Informationen finden Sie unter So verwenden Sie einen Alias im Bucket-Stil für den Object Lambda Access Point Ihres S3-Buckets.
- Bucket-Namen müssen in allen AWS-Konten innerhalb AWS-Regionen einer Partition eindeutig sein. Eine Partition ist eine Gruppierung von Regionen. hat AWS derzeit drei Partitionen: `aws` (Standardregionen), `aws-cn` (China-Regionen) und `aws-us-gov` (AWS GovCloud (US)).
- Ein Bucket-Name kann erst von einem anderen AWS-Konto in derselben Partition verwendet werden, wenn der Bucket gelöscht wurde.
- Buckets, die mit Amazon S3 Transfer Acceleration verwendet werden, können keine Punkte (.) in ihren Namen haben. Weitere Informationen zu Transfer Acceleration finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.

Aus Gründen der besten Kompatibilität empfehlen wir, Punkte (.) in Bucket-Namen zu vermeiden, mit Ausnahme von Buckets, die nur für statisches Website-Hosting verwendet werden. Wenn Sie Punkte in den Namen eines Buckets aufnehmen, können Sie die virtual-host-style Adressierung nicht über HTTPS verwenden, es sei denn, Sie führen Ihre eigene Zertifikatvalidierung durch. Dies liegt daran, dass die Sicherheitszertifikate, die für das virtuelle Hosten von Buckets verwendet werden, nicht für Buckets mit Punkten in ihren Namen funktionieren.

Diese Einschränkung wirkt sich nicht auf Buckets aus, die für das Hosten statischer Websites verwendet werden, da das Hosten von statischen Websites nur über HTTP verfügbar ist. Weitere Informationen zur virtual-host-style Adressierung finden Sie unter Virtuelles Hosting bei Buckets. Weitere Hinweise zum Hosten statischer Websites finden Sie unter Hosten einer statischen Website mit Amazon S3.

> **Note**
>
> Vor dem 1. März 2018 konnten Buckets, die in der Region USA Ost (Nord-Virginia) erstellt wurden, Namen mit bis zu 255 Zeichen und mit Großbuchstaben und Unterstrichen haben.

Ab dem 1. März 2018 müssen neue Buckets in USA Ost (Nord-Virginia) den gleichen Regeln entsprechen, die in allen anderen Regionen angewendet werden.

Informationen zu Objektschlüsselnamen finden Sie unter Erstellen von Objektschlüsselnamen.

Beispiel für Namen von Buckets für allgemeine Zwecke

Dies sind Beispiele für gültige Namen von Buckets. Sie befolgen die Empfehlungen für die Benennung von Buckets für allgemeine Zwecke:

- `docexamplebucket1`
- `log-delivery-march-2020`
- `my-hosted-content`

Die folgenden Beispiel-Bucket-Namen sind gültig, aber nicht für andere Verwendungszwecke als statisches Website-Hosting empfohlen:

- `docexamplewebsite.com`
- `www.docexamplewebsite.com`
- `my.example.s3.bucket`

Die folgenden Beispiel-Bucket-Namen sind ungültig:

- `doc_example_bucket` (enthält Unterstriche)
- `DocExampleBucket` (enthält Großbuchstaben)
- `doc-example-bucket-` (endet mit einem Bindestrich)

Regeln für die Benennung von Verzeichnis-Buckets

Die Namen von Verzeichnis-Buckets müssen folgende Kriterien erfüllen:

- innerhalb der ausgewählten AWS-Region und Availability Zone eindeutig sein.
- Nicht mehr als 3-63 Zeichen lang sein, einschließlich des Suffixes.
- Nur aus Kleinbuchstaben, Zahlen und Bindestrichen bestehen.
- Muss mit einer Zahl oder einem Buchstaben beginnen und enden.

- Muss das folgende Suffix enthalten: `--azid--x-s3`.

> **Note**
>
> Wenn Sie einen Verzeichnis-Bucket mit der Konsole erstellen, wird dem von Ihnen angegebenen Basisnamen automatisch ein Suffix hinzugefügt. Dieses Suffix enthält die ID der Availability Zone, die Sie ausgewählt haben.
> Wenn Sie einen Verzeichnis-Bucket mit einer API erstellen, müssen Sie das vollständige Suffix, einschließlich der Availability Zone-ID, in Ihrer Anfrage angeben. Eine Liste der Availability Zone-IDs finden Sie unter Availability Zones und Regionen bei S3 Express One Zone.

Zugreifen auf einen Amazon-S3-Bucket und Auflisten des Buckets

Sie können verschiedene Tools verwenden, um auf Amazon-S3-Buckets zuzugreifen und sie aufzulisten. Gehen Sie die folgenden Tools durch und finden Sie heraus, welcher Ansatz für Ihren Anwendungsfall geeignet ist:

- Amazon-S3-Konsole: Mit der Amazon-S3-Konsole können Sie problemlos auf einen Bucket zugreifen und seine Eigenschaften ändern. Mit der Benutzeroberfläche der Konsole können Sie zudem fast alle Bucket-Operationen ausführen, ohne Code schreiben zu müssen.
- AWS CLI: Wenn Sie auf mehrere Buckets zugreifen müssen, können Sie Zeit sparen, indem Sie die AWS Command Line Interface (AWS CLI) verwenden, um allgemeine und sich wiederholende Aufgaben zu automatisieren. Die Skriptfähigkeit und die Wiederholbarkeit gängiger Aktionen sind wichtige Aspekte, die bei der Skalierung von Unternehmen zu berücksichtigen sind. Weitere Informationen finden Sie unter Entwickeln mit Amazon S3 über die AWS CLI.
- Amazon-S3-REST-API: Mithilfe der Amazon-S3-REST-API können Sie eigene Programme schreiben und programmgesteuert auf Buckets zugreifen. Amazon S3 unterstützt eine API-Architektur, in der die Buckets und Objekte Ressourcen sind. Sie besitzen alle eine Ressourcen-URI, die die jeweilige Ressource eindeutig identifiziert. Weitere Informationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der REST-API.

Je nach Anwendungsfall des Amazon-S3-Buckets werden verschiedene Methoden für den Zugriff auf die zugrunde liegenden Daten in den Buckets empfohlen. Die folgende Liste enthält gängige Anwendungsfälle für den Zugriff auf Daten.

- Statische Websites – Sie können in Amazon S3 eine statische Website hosten. Bei diesem Anwendungsfall lässt sich der S3-Bucket so konfigurieren, dass er sich wie eine Website verhält. Ein Beispiel für Schritte zum Hosten einer Website auf Amazon S3 finden Sie unter Tutorial: Konfigurieren einer statischen Website auf Amazon S3.

  Um eine statische Website mit aktivierten Sicherheitseinstellungen wie Block Public Access zu hosten, empfehlen wir, Amazon CloudFront mit Origin Access Control (OAC) zu verwenden und zusätzliche Sicherheitsheader wie HTTPS zu implementieren. Weitere Informationen finden Sie unter Erste Schritte mit einer sicheren statischen Website.

  > **Note**
  >
  > Amazon S3 unterstützt für den Zugriff auf statische Websites sowohl URLs im virtuellen Hosting-Format als auch im Pfadformat. Da der Zugriff auf Buckets über URLs im Pfad-Stil und im Stil des virtuellen Hostings möglich ist, empfehlen wir Ihnen, Buckets mit DNS-konformen Bucket-Namen zu erstellen. Weitere Informationen finden Sie unter Beschränkungen und Einschränkungen von Buckets.

- Freigegebene Datensätze – Bei der Skalierung auf Amazon S3 wird üblicherweise ein Multi-Tenant-Modell verwendet, bei dem Sie verschiedenen Endkunden oder Geschäftseinheiten eindeutige Präfixe innerhalb eines freigegebenen Buckets zuweisen. Durch die Verwendung von Amazon S3 Access Points können Sie eine umfangreiche Bucket-Richtlinie in separate, diskrete Zugangspunkt-Richtlinien für jede Anwendung unterteilen, die auf den freigegebenen Datensatz zugreifen muss. Mit diesem Ansatz ist es einfacher, sich auf die Erstellung der richtigen Zugriffsrichtlinie für eine Anwendung zu konzentrieren, ohne die Aktivitäten anderer Anwendungen innerhalb des freigegebenen Datensatzes zu beeinflussen. Weitere Informationen finden Sie unter Verwalten des Datenzugriffs mit Amazon S3-Zugangspunkten.
- Workload mit hohem Durchsatz – Mountpoint für Amazon S3 ist ein Open-Source-Dateiclient mit hohem Durchsatz zum Mounten eines Amazon-S3-Bucket als lokales Dateisystem. Mit Mountpoint können Anwendungen über Dateisystem-Operationen wie Öffnen und Lesen auf Objekte zugreifen, die in Amazon S3 gespeichert sind. Mountpoint übersetzt diese Operationen automatisch in API-Aufrufe für S3-Objekte, sodass Ihre Anwendungen über eine Dateischnittstelle auf den elastischen Speicher und den Durchsatz von Amazon S3 zugreifen können. Weitere Informationen finden Sie unter Arbeiten mit Mountpoint für Amazon S3.
- Multi-Regions-Anwendungen – Amazon S3 Multi-Region Access Points stellen einen globalen Endpunkt bereit, mit dem Anwendungen Anforderungen von S3-Buckets ausführen können, die sich in mehreren AWS-Regionen-Regionen befinden. Sie können Multi-Regions Access Points

verwenden, um Multi-Regions-Anwendungen mit derselben Architektur zu erstellen, die in einer einzelnen Region verwendet wird, und diese Anwendungen dann überall auf der Welt ausführen. Anstatt Anforderungen über das öffentliche Internet zu senden, bieten Multi-Region Access Points integrierte Netzwerkausfallsicherheit mit Beschleunigung internetbasierter Anforderungen an Amazon S3. Weitere Informationen finden Sie unter Multi-Regions-Zugriffspunkte in Amazon S3.

- Erstellen neuer Anwendungen – Sie können die - AWS SDKs bei der Entwicklung von Anwendungen mit Amazon S3 verwenden. Die - AWS SDKs vereinfachen Ihre Programmieraufgaben, indem sie die zugrunde liegende Amazon S3-REST-API umschließen. Um verbundene mobile und Webanwendungen zu erstellen, können Sie die AWS Mobile SDKs und die AWS Amplify JavaScript Bibliothek verwenden. Weitere Informationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.
- Secure Shell (SSH) File Transfer Protocol (SFTP) – Wenn Sie versuchen, vertrauliche Daten sicher über das Internet zu übertragen, können Sie einen SFTP-fähigen Server mit Ihrem Amazon S3-Bucket verwenden. AWS SFTP ist ein Netzwerkprotokoll, das die volle Sicherheits- und Authentifizierungsfunktionalität von SSH unterstützt. Mit diesem Protokoll haben Sie eine genaue Kontrolle über Benutzeridentität, Berechtigungen und Schlüssel. Sie können den Zugriff auch mithilfe von IAM-Richtlinien verwalten. Wenn Sie dem Amazon-S3-Bucket einen SFTP-fähigen Server zuordnen möchten, müssen Sie zunächst einen SFTP-fähigen Server erstellen. Anschließend richten Sie Benutzerkonten ein und verknüpfen den Server mit einem Amazon-S3-Bucket. Eine exemplarische Vorgehensweise für diesen Prozess finden Sie unter AWS Transfer for SFTP – Fully Managed SFTP Service for Amazon S3 in AWS Blogs .

Auflisten eines Buckets

Zum Auflisten sämtlicher Buckets benötigen Sie die Genehmigung `s3:ListAllMyBuckets`. Um auf einen Bucket zuzugreifen, stellen Sie sicher, dass Sie auch die erforderlichen AWS Identity and Access Management (IAM)-Berechtigungen zum Auflisten des Inhalts des angegebenen Buckets erhalten. Ein Beispiel für eine Bucket-Richtlinie, die den Zugriff auf einen S3-Bucket gewährt, finden Sie unter Einem IAM-Benutzer den Zugriff auf einen Ihrer Buckets erlauben. Wenn der Fehler „HTTP Access Denied (403 Forbidden)“ angezeigt wird, finden Sie weitere Informationen unter Bucket-Richtlinien und IAM-Richtlinien.

Sie können Ihren Bucket mithilfe der Amazon S3-Konsole, der AWS CLI oder der AWS SDKs auflisten.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des gewünschten Buckets aus.

Verwenden der AWS CLI

Um mit AWS CLI der auf einen S3-Bucket zuzugreifen oder eine Liste von S3-Buckets zu generieren, verwenden Sie den `ls` Befehl . Sollen alle Objekte im Bucket aufgelistet werden, müssen Sie über die Berechtigung `s3:ListBucket` verfügen.

Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *DOC-EXAMPLE-BUCKET1* durch den Namen des Buckets.

```
$ aws s3 ls s3://DOC-EXAMPLE-BUCKET1
```

Mit dem folgenden Beispielbefehl werden alle Amazon-S3-Buckets in Ihrem Konto aufgelistet.

```
$ aws s3 ls
```

Weitere Informationen finden Sie unter Auflisten von Buckets und Objekten.

Verwenden der AWS SDKs

Sie können auch mithilfe der API-Operation ListBuckets auf einen Amazon-S3-Bucket zugreifen. Beispiele für die Verwendung dieser Operation mit verschiedenen - AWS SDKs finden Sie unter Auflisten von Amazon S3-Buckets mithilfe eines - AWS SDK.

Erstellen eines Buckets

Um Ihre Daten auf Amazon S3 hochzuladen, müssen Sie zuerst einen Amazon-S3-Bucket in einer der AWS-Regionen erstellen. Wenn Sie einen Bucket erstellen, müssen Sie einen Bucket-Namen und eine Region auswählen. Sie können optional andere Speicherverwaltungsoptionen für den Bucket auswählen. Der Name eines erstellten Buckets oder Region kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

Das AWS-Konto , das den Bucket erstellt, besitzt ihn. Sie können beliebig viele Objekte in den Bucket hochladen. Standardmäßig können Sie bis zu 100 Buckets in jedem Ihrer erstellen AWS-Konten. Wenn Sie weitere Buckets benötigen, können Sie das Konto-Bucket-Limit auf maximal 1 000 Buckets erhöhen, indem Sie eine Service Limit-Erhöhung senden. Informationen zum Einreichen einer Bucket-Limit-Erhöhung finden Sie unter AWS-Service -Quotas in der Allgemeinen AWS -Referenz. Sie können in einem Bucket beliebig viele Objekte speichern.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch Zugriffssteuerungslisten (ACLs) deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer jedes Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Richtlinien.

Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) ist die Grundebene der Verschlüsselungskonfiguration für jeden Bucket in Amazon S3. Alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden automatisch mit SSE-S3 als Grundebene für die Verschlüsselungsstufe verschlüsselt. Wenn Sie eine andere Standardverschlüsselung verwenden möchten, können Sie auch eine serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder vom Kunden bereitgestellten Schlüsseln (SSE-C) für Ihre Daten festlegen. Weitere Informationen finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

Sie können die Amazon S3-Konsole, Amazon S3-APIs oder - AWS SDKs verwenden AWS CLI, um einen Bucket zu erstellen. Weitere Informationen zu den erforderlichen Berechtigungen zum Erstellen eines Buckets finden Sie unter CreateBucket in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Anschließend wird die Seite Bucket erstellen geöffnet.

4. Geben Sie unter Bucket Name (Bucket-Name) einen Namen für den Bucket ein.

   Der Bucket-Name ...:

   - Muss innerhalb einer Partition einzigartig sein. Eine Partition ist eine Gruppierung von Regionen. AWS verfügt derzeit über drei Partitionen: `aws` (Standardregionen), `aws-cn` (China-Regionen) und `aws-us-gov` (AWS GovCloud (US) Regions).
   - zwischen 3 und 63 Zeichen lang sein,
   - Darf nur aus Kleinbuchstaben, Zahlen, Punkten (.) und Bindestrichen (-) bestehen. Aus Gründen der besten Kompatibilität empfehlen wir, Punkte (.) in Bucket-Namen zu vermeiden, mit Ausnahme von Buckets, die nur für statisches Website-Hosting verwendet werden.
   - Muss mit einer Zahl oder einem Buchstaben beginnen und enden.

   Der Name eines einmal erstellter Buckets kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

   > **Important**
   >
   > Vermeiden Sie, vertrauliche Informationen, wie Kontonummern, in den Bucket-Namen einzubeziehen. Der Bucket-Name wird in der URL angezeigt, die auf die Objekte im Bucket verweist.

5. Wählen Sie für Region die aus, AWS-Region in der sich der Bucket befinden soll.

   Wählen Sie eine Region in der Nähe aus, um Latenz und Kosten gering zu halten und behördliche Vorschriften zu erfüllen. In einer Region gespeicherte Objekte verbleiben so lange in der Region, bis sie explizit in eine andere Region verschoben werden. Eine Liste von Amazon S3 AWS-Regionen finden Sie unter -AWS-Service Endpunkte im Allgemeine Amazon Web Services-Referenz.

6. Wählen Sie unter Object Ownership eine der folgenden Einstellungen aus, um ACLs zu deaktivieren oder zu aktivieren und den Besitz von Objekten zu steuern, die in Ihren Bucket hochgeladen wurden:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Zugriffsberechtigungen für Daten im S3-Bucket. Der Bucket verwendet ausschließlich Richtlinien, um die Zugriffssteuerung zu definieren.

  Standardmäßig sind ACLs deaktiviert. Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.

  Wenn Sie die Einstellung Bucket-Eigentümer bevorzugt anwenden, damit alle Amazon-S3-Uploads die von `bucket-owner-full-control` vordefinierte ACL enthalten, können Sie eine Bucket-Richtlinie hinzufügen, die nur Objekt-Uploads zulässt, die diese ACL verwenden.
- Object Writer – Das AWS-Konto , das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

> **Note**
>
> Die Standardeinstellung ist Bucket-Eigentümer erzwungen. Um die Standardeinstellung anzuwenden und ACLs deaktiviert zu lassen, ist nur die `s3:CreateBucket`-Berechtigung erforderlich. Sie müssen über die `s3:PutBucketOwnershipControls`-Berechtigung verfügen, um ACLs zu aktivieren.

7. Wählen Sie unter Einstellungen "Öffentlichen Zugriff beschränken" für diesen Bucket die Einstellungen zum Beschränken des öffentlichen Zugriffs aus, die Sie auf den Bucket anwenden möchten.

Alle vier Einstellungen zum Blockieren des öffentlichen Zugriffs sind standardmäßig aktiviert. Es wird empfohlen, alle Einstellungen aktiviert zu lassen, es sei denn, Sie wissen, dass Sie eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren müssen. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

> **Note**
>
> Zum Aktivieren aller Einstellungen zum Blockieren des öffentlichen Zugriffs ist nur die `s3:CreateBucket`-Berechtigung erforderlich. Wenn Sie eine der Einstellungen zum Blockieren des öffentlichen Zugriffs deaktivieren möchten, benötigen Sie die `s3:PutBucketPublicAccessBlock`-Berechtigung.

8. (Optional) Unter Bucket Versioning (Bucket-Versionsverwaltung) können Sie auswählen, ob Sie Varianten von Objekten in Ihrem Bucket beibehalten möchten. Weitere Informationen über das Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

   Wenn Sie die Versionsverwaltung in Ihrem Bucket deaktivieren oder aktivieren möchten, wählen Sie entweder Disable (Deaktivieren) oder Enable (Aktivieren) aus.

9. (Optional) Unter Tags können Sie auswählen, ob Sie Ihrem Bucket Tags hinzufügen möchten. Tags sind Schlüssel-Wert-Paare, die zur Kategorisierung von Speicher verwendet werden.

   Wenn Sie ein Bucket-Tag hinzuzufügen, geben Sie einen Key (Schlüssel) und optional einen Value (Wert) ein. Wählen Sie dann Add Tag (Tag hinzufügen) aus.

10. Wählen Sie unter Default encryption (Standard-Verschlüsselung) Edit (Bearbeiten) aus.

11. Wählen Sie eine der folgenden Optionen unter Verschlüsselungstyp aus, um die Standardverschlüsselung zu konfigurieren:

    - Von Amazon S3 verwalteter Schlüssel (SSE-S3)
    - AWS Key Management Service -Schlüssel (SSE-KMS)

    > **Important**
    >
    > Wenn Sie die Option SSE-KMS für die Standardverschlüsselung verwenden, unterliegen Sie den Kontigenten der Anforderungen pro Sekunde (RPS) von AWS KMS. Weitere Informationen zu AWS KMS Kontingenten und zum Anfordern einer

Kontingenterhöhung finden Sie unter Kontingente im AWS Key Management Service - Entwicklerhandbuch.

Buckets und neue Objekte werden mit serverseitiger Verschlüsselung verschlüsselt. Dabei ist ein Von Amazon S3 verwalteter Schlüssel die Grundebene der Verschlüsselungskonfiguration. Weitere Informationen zur Standardverschlüsselung finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

Weitere Informationen zur Datenverschlüsselung mit der serverseitigen Amazon-S3-Verschlüsselung finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

12. Wenn Sie AWS Key Management Service -Schlüssel (SSE-KMS) ausgewählt haben, gehen Sie wie folgt vor:

    a. Geben Sie unter AWS KMS -Schlüssel Ihren KMS-Schlüssel auf eine der folgenden Arten an:

    - Um aus einer Liste der verfügbaren KMS-Schlüssel auszuwählen, wählen Sie Aus Ihrem AWS KMS keys auswählen und wählen Sie Ihren KMS-Schlüssel aus der Liste der verfügbaren Schlüssel aus.

      Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS -Schlüssel im Entwicklerhandbuch zu AWS Key Management Service .

    - Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS key -ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.

    - Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

      Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

> **Important**
>
> Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket verfügbar sind. Die Amazon-S3-Konsole führt nur die ersten 100 KMS-Schlüssel auf, die in derselben Region wie der Bucket verfügbar sind. Wenn Sie einen KMS-Schlüssel verwenden möchten, der nicht aufgeführt ist, müssen Sie den KMS-Schlüssel-ARN eingeben. Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen und Sie müssen den KMS-Schlüssel-ARN eingeben. Weitere Informationen zu kontoübergreifenden Berechtigungen für KMS-Schlüssel finden Sie unter Erstellen von KMS-Schlüsseln, die von anderen Konten verwendet werden können im Entwicklerhandbuch zu AWS Key Management Service . Weitere Informationen zu SSE-KMS finden Sie unter Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS).
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung und keine asymmetrischen KMS-Schlüssel. Weitere Informationen finden Sie unter Erkennen von symmetrischen und asymmetrischen KMS-Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service .

Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch. Weitere Informationen zur Verwendung von AWS KMS mit Amazon S3 finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

b. Wenn Sie Ihren Bucket für die Verwendung der Standardverschlüsselung mit SSE-KMS konfigurieren, können Sie auch S3-Bucket-Schlüssel aktivieren. S3-Bucket-Schlüssel senken die Verschlüsselungskosten, indem der Anforderungsverkehr von Amazon S3 zu verringert wird AWS KMS. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

   Um S3-Bucket-Schlüssel zu verwenden, wählen Sie unter Bucket Key (Bucket-Schlüssel) die Option Enable (Aktivieren).

13. (Optional) Wenn Sie die S3-Objektsperre aktivieren möchten, gehen Sie wie folgt vor:

a. Wählen Sie Erweiterte Einstellungen aus.

> **Important**
>
> Durch Aktivieren der Objektsperre wird auch die Versioning für den Bucket aktiviert. Nach dem Aktivieren müssen Sie die Standardeinstellungen für die Objektsperre im Hinblick auf die (rechtliche) Aufbewahrung konfigurieren, um neue Objekte vor dem Löschen oder Überschreiben zu schützen.

b. Wenn Sie die Objektsperre aktivieren möchten, wählen Sie Enable (Aktivieren) aus, lesen Sie die angezeigte Warnung und bestätigen Sie sie.

Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

> **Note**
>
> Wenn Sie einen Bucket mit aktivierter Objektsperre erstellen möchten, benötigen Sie die folgenden Berechtigungen: `s3:CreateBucket`, `s3:PutBucketVersioning` und `s3:PutBucketObjectLockConfiguration`.

14. Wählen Sie Bucket erstellen aus.

Verwenden der AWS SDKs

Wenn Sie die - AWS SDKs verwenden, um einen Bucket zu erstellen, müssen Sie einen Client erstellen und dann den Client verwenden, um eine Anforderung zum Erstellen eines Buckets zu senden. Als bewährte Methode sollten Sie Ihren Client und Ihren Bucket in derselben AWS-Region erstellen. Wenn Sie beim Erstellen eines Clients oder Buckets keine Region angeben, verwendet Amazon S3 die Standardregion USA Ost (Nord-Virginia). Wenn Sie die Bucket-Erstellung auf eine bestimmte AWS-Region beschränken möchten, verwenden Sie dazu den Bedingungsschlüssel `LocationConstraint`.

Um einen Client für den Zugriff auf einen Dual-Stack-Endpunkt zu erstellen, müssen Sie eine AWS-Region angeben. Weitere Informationen finden Sie unter Dual-Stack-Endpunkte. Eine Liste der verfügbaren finden AWS-Regionen Sie unter Regionen und Endpunkte im Allgemeine AWS-Referenz.

Wenn Sie einen Client erstellen, wird die Region dem regionsspezifischen Endpunkt zugeordnet. Der Client verwendet diesen Endpunkt für die Kommunikation mit Amazon S3: `s3.region.amazonaws.com`. Wenn Ihre Region nach dem 20. März 2019 gestartet wurde, müssen sich Ihr Client und Ihr Bucket in derselben Region befinden. Sie können jedoch einen Client in der Region USA Ost (Nord-Virginia) verwenden, um einen Bucket in einer beliebigen Region zu erstellen, die vor dem 20. März 2019 gestartet wurde. Weitere Informationen finden Sie unter Legacy-Endpunkte.

Diese AWS SDK-Codebeispiele führen die folgenden Aufgaben aus:

- Erstellen eines Clients durch explizite Angabe einer AWS-Region – Im Beispiel verwendet der Client den Endpunkt `s3.us-west-2.amazonaws.com`, um mit Amazon S3 zu kommunizieren. Sie können eine beliebige AWS-Region angeben. Eine Liste von AWS-Regionen finden Sie unter Regionen und Endpunkte in der AWS Allgemeinen Referenz zu .
- Senden einer Bucket-Erstellungs-Anfrage durch Angabe eines Bucket-Namens – Der Client sendet eine Anfrage an Amazon S3, um den Bucket in der Region zu erstellen, in der Sie einen Client erstellt haben.
- Abrufen von Informationen zum Standort des Buckets – Amazon S3 speichert Informationen zum Standort des Buckets in der Subressource Standort, die dem Bucket zugeordnet ist.

Java

Dieses Beispiel zeigt, wie ein Amazon-S3-Bucket mit dem erstellt wird AWS SDK for Java. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CreateBucketRequest;
import com.amazonaws.services.s3.model.GetBucketLocationRequest;

import java.io.IOException;

public class CreateBucket2 {
```

```
    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            if (!s3Client.doesBucketExistV2(bucketName)) {
                // Because the CreateBucketRequest object doesn't specify a region,
 the
                // bucket is created in the region specified in the client.
                s3Client.createBucket(new CreateBucketRequest(bucketName));

                // Verify that the bucket was created by retrieving it and checking
 its
                // location.
                String bucketLocation = s3Client.getBucketLocation(new
 GetBucketLocationRequest(bucketName));
                System.out.println("Bucket location: " + bucketLocation);
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.S3.Util;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CreateBucketTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CreateBucketAsync().Wait();
        }

        static async Task CreateBucketAsync()
        {
            try
            {
                if (!(await AmazonS3Util.DoesS3BucketExistAsync(s3Client,
 bucketName)))
                {
                    var putBucketRequest = new PutBucketRequest
                    {
                        BucketName = bucketName,
                        UseClientRegion = true
                    };

                    PutBucketResponse putBucketResponse = await
 s3Client.PutBucketAsync(putBucketRequest);
                }
                // Retrieve the bucket location.
                string bucketLocation = await FindBucketLocationAsync(s3Client);
            }
            catch (AmazonS3Exception e)
```

```
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
        static async Task<string> FindBucketLocationAsync(IAmazonS3 client)
        {
            string bucketLocation;
            var request = new GetBucketLocationRequest()
            {
                BucketName = bucketName
            };
            GetBucketLocationResponse response = await
 client.GetBucketLocationAsync(request);
            bucketLocation = response.Location.ToString();
            return bucketLocation;
        }
    }
}
```

Ruby

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwenden von AWS SDK for Ruby – Version 3.

Example

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketCreateWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An Amazon S3 bucket initialized with a name.
 This is a client-side object until
  #                                 create is called.
  def initialize(bucket)
    @bucket = bucket
  end
```

```
  # Creates an Amazon S3 bucket in the specified AWS Region.
  #
  # @param region [String] The Region where the bucket is created.
  # @return [Boolean] True when the bucket is created; otherwise, false.
  def create?(region)
    @bucket.create(create_bucket_configuration: { location_constraint: region })
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't create bucket. Here's why: #{e.message}"
    false
  end

  # Gets the Region where the bucket is located.
  #
  # @return [String] The location of the bucket.
  def location
    if @bucket.nil?
      "None. You must create a bucket before you can get its location!"
    else
      @bucket.client.get_bucket_location(bucket: @bucket.name).location_constraint
    end
  rescue Aws::Errors::ServiceError => e
    "Couldn't get the location of #{@bucket.name}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  region = "us-west-2"
  wrapper = BucketCreateWrapper.new(Aws::S3::Bucket.new("doc-example-bucket-
#{Random.uuid}"))
  return unless wrapper.create?(region)

  puts "Created bucket #{wrapper.bucket.name}."
  puts "Your bucket's region is: #{wrapper.location}"
end

run_demo if $PROGRAM_NAME == __FILE__
```

Verwenden der AWS CLI

Sie können auch die AWS Command Line Interface (AWS CLI) verwenden, um einen S3-Bucket zu erstellen. Weitere Informationen finden Sie unter create-bucket in der AWS CLI -Befehlsreferenz.

Weitere Informationen zu finden AWS CLI Sie unter Was ist AWS Command Line Interface? im AWS Command Line Interface -Benutzerhandbuch.

Anzeigen der Eigenschaften eines S3-Buckets

Sie können die Eigenschaften für einen Amazon S3-Bucket anzeigen und konfigurieren, einschließlich Einstellungen für Versioning-Steuerung, Tags, Standardverschlüsselung, Protokollierung, Benachrichtigungen und mehr.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets) den Namen des Buckets aus, dessen Eigenschaften Sie anzeigen wollen.
3. Wählen Sie Properties (Eigenschaften).
4. Auf der Seite Properties (Eigenschaften) können Sie die folgenden Eigenschaften für den Bucket konfigurieren.

   - Bucket Versioning (Bucket-Versionssteuerung) – Behalten Sie mehrere Versionen eines Objekts in einem Bucket bei, indem Sie die Versionssteuerung verwenden. Für einen neuen Bucket ist das Versioning standardmäßig deaktiviert. Weitere Informationen über die Aktivierung des Versionings finden Sie unter Aktivieren des Versioning für Buckets.
   - Tags – Mit der AWS Kostenzuordnung können Sie Bucket-Tags verwenden, um die Abrechnung für Ihre Verwendung eines Buckets zu kommentieren. Ein Tag ist ein Schlüssel-Wert-Paar, das eine Bezeichnung repräsentiert, die Sie einem Bucket zuweisen. Um Tags hinzuzufügen, wählen Sie Tags und dann Add tag (Tag hinzufügen). Weitere Informationen finden Sie unter Verwenden von Kostenzuordnungs-Markierungen für S3-Buckets.
   - Default encryption (Standardverschlüsselung) – Die Aktivierung der Standardverschlüsselung bietet Ihnen eine automatische serverseitige Verschlüsselung. Amazon S3 verschlüsselt ein Objekt, bevor es auf einer Festplatte gespeichert wird, und entschlüsselt das Objekt

beim Herunterladen. Weitere Informationen finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

- Server access logging (Server-Zugriffsprotokollierung) – Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anfragen, die an Ihren Bucket gestellt wurden. Standardmäßig erfasst Amazon S3 keine Server-Zugriffsprotokolle. Weitere Informationen zur Aktivierung der Server-Zugriffsprotokollierung finden Sie unter Aktivieren Sie die Amazon-S3-Server-Zugriffsprotokollierung
- -AWS CloudTrail Datenereignisse – Wird verwendet CloudTrail , um Datenereignisse zu protokollieren. Standardmäßig werden Datenereignisse nicht von den Trails protokolliert. Für Datenereignisse werden zusätzliche Gebühren fällig. Weitere Informationen finden Sie unter Protokollierung von Datenereignissen für Trails im AWS CloudTrail -Benutzerhandbuch.
- Event notifications (Ereignisbenachrichtigungen)Ereignisse – Sie können bestimmte Amazon S3-Bucket-Ereignisse aktivieren, um eine Benachrichtigungsmeldung an ein Ziel zu senden, wenn das Ereignis auftritt. Um Ereignisse zu aktivieren, wählen Sie Create event notification (Ereignisbenachrichtigung erstellen) und geben Sie dann die Einstellungen an, die Sie verwenden möchten. Weitere Informationen finden Sie unter Aktivieren und Konfigurieren von Ereignis-Benachrichtigungen mit der Amazon-S3-Konsole.
- Transfer Acceleration – Ermöglicht schnelle, einfache und sichere Übertragungen von Dateien zwischen Ihrem Client und einem S3-Bucket über große Entfernungen. Weitere Informationen zur Aktivierung der Transfer Acceleration finden Sie unter Aktivieren und Verwenden von S3 Transfer Acceleration.
- Object Lock – Mit der S3-Objektsperre können Sie für einen festen Zeitraum oder auf unbegrenzte Zeit verhindern, dass ein Objekt gelöscht oder überschrieben wird. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.
- Requester Pays (Zahlung durch den Anforderer) – Sie können die Zahlung durch den Anforderer aktivieren, sodass der Anforderer (anstelle des Bucket-Eigentümers) für die Anforderungen und Datenübertragungen zahlt. Weitere Informationen finden Sie unter Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung.
- Static website hosting (Hosting statischer Websites) – Sie können in Amazon S3 eine statische Website hosten. Um das Hosting einer statischen Website zu aktivieren, wählen Sie Static website hosting (Hosting statischer Websites), und geben Sie die gewünschten Einstellungen an. Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.

Verwenden der AWS CLI

Sie können auch die AWS Command Line Interface (AWS CLI) verwenden, um die Eigenschaften für einen S3-Bucket anzuzeigen. Weitere Informationen finden Sie unter den folgenden Befehlen in der AWS CLI -Befehlsreferenz.

- get-bucket-tagging
- get-bucket-versioning
- get-bucket-encryption
- get-bucket-notification-configuration
- get-bucket-logging

Weitere Informationen zu finden AWS CLI Sie unter Was ist AWS Command Line Interface? im AWS Command Line Interface -Benutzerhandbuch.

Leeren eines Buckets

Sie können den Inhalt eines Buckets mit der Amazon S3-Konsole, AWS SDKs oder AWS Command Line Interface () leeren AWS CLI. Wenn Sie einen Bucket leeren, löschen Sie alle Objekte, behalten aber den Bucket. Nachdem Sie einen Bucket geleert haben, kann es nicht rückgängig gemacht werden. Dem Bucket während der Löschaktion hinzugefügte Objekte werden möglicherweise gelöscht. Alle Objekte (einschließlich aller Objektversionen und Löschmarkierungen) im Bucket müssen gelöscht werden, bevor der Bucket selbst gelöscht werden kann.

Wenn Sie einen Bucket mit aktivierter oder ausgesetzter S3-Versionsverwaltung leeren, werden alle Versionen aller Objekte im Bucket gelöscht. Weitere Informationen finden Sie unter Arbeiten mit Objekten in einem versioning-fähigen Bucket.

Sie können auch eine Lebenszyklus-Konfiguration für einen Bucket angeben, sodass Objekte ablaufen, damit Amazon S3 sie löschen kann. Weitere Informationen finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket. Um einen großen Bucket zu leeren, empfehlen wir, eine S3-Lifecycle-Konfigurationsregel zu verwenden. Der Ablauf des Lebenszyklus ist ein asynchroner Prozess, daher kann es einige Tage dauern, bis die Regel ausgeführt wird und der Bucket leer ist. Nachdem Amazon S3 die Regel zum ersten Mal ausführt, werden alle Objekte, die für den Ablauf in Frage kommen, zum Löschen markiert. Die Objekte, die zum Löschen markiert sind, werden Ihnen nicht mehr in Rechnung gestellt. Weitere Informationen finden Sie unter Wie leere ich einen Amazon-S3-Bucket mithilfe einer Lebenszykluskonfigurationsregel?.

Verwenden der S3-Konsole

Sie können die Amazon-S3-Konsole verwenden, um einen Bucket zu leeren, der alle Objekte im Bucket löscht, ohne den Bucket zu löschen.

Einen S3-Bucket leeren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket Name (Bucket-Name) die Option neben dem Namen des Buckets aus, den Sie leeren möchten. Wählen Sie anschließend Empty (Leeren) aus.
3. Bestätigen Sie auf der Seite Empty bucket (Leerer Bucket), dass Sie den Bucket leeren möchten, indem Sie den Bucket-Namen in das Textfeld eingeben, und wählen Sie dann Empty (Leeren).
4. Überwachen Sie den Fortschritt des Vorgangs zum Entleeren von Buckets auf der Seite Empty bucket: Status (Bucket entleeren: Status).

Verwenden der AWS CLI

Sie können einen Bucket AWS CLI nur mit dem leeren, wenn für den Bucket die Bucket-Versionsverwaltung nicht aktiviert ist. Wenn das Versioning nicht aktiviert ist, können Sie den AWS CLI Befehl `rm` (remove) mit dem `--recursive` Parameter verwenden, um den Bucket zu leeren (oder eine Untermenge von Objekten mit einem bestimmten Schlüsselnamenpräfix zu entfernen).

Der folgende `rm`-Befehl entfernt Objekte mit dem Schlüsselnamenpräfix `doc`, z. B. `doc/doc1` und `doc/doc2`.

```
$ aws s3 rm s3://bucket-name/doc --recursive
```

Verwenden Sie den folgenden Befehl, um alle Objekte zu entfernen, ohne ein Präfix anzugeben.

```
$ aws s3 rm s3://bucket-name --recursive
```

Weitere Informationen finden Sie unter Using High-Level S3 Commands with the AWS CLI(Verwenden von hochrangigen S3-Befehlen mit der CLI) im AWS Command Line Interface - Benutzerhandbuch.

> **Note**
>
> Sie können keine Objekte aus einem Bucket entfernen, für den die Versionierung aktiviert ist. Amazon S3 fügt eine Löschmarkierung hinzu, wenn Sie ein Objekt löschen, was dieser Befehl tut. Weitere Informationen über die S3-Bucket-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

Verwenden der AWS SDKs

Sie können die - AWS SDKs verwenden, um einen Bucket zu leeren oder eine Untermenge von Objekten mit einem bestimmten Schlüsselnamenpräfix zu entfernen.

Ein Beispiel für das Leeren eines Buckets mit AWS SDK for Java finden Sie unter Löschen eines Buckets. Der Code löscht alle Objekte, unabhängig davon, ob für den Bucket das Versioning aktiviert ist. Anschließend löscht er den Bucket. Um den Bucket nur zu leeren, stellen Sie sicher, dass Sie die Anweisung entfernen, die den Bucket löscht.

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Tools für Amazon Web Services.

Verwenden einer Lebenszyklus-Konfiguration

Um einen großen Bucket zu leeren, empfehlen wir, eine S3-Lifecycle-Konfigurationsregel zu verwenden. Der Ablauf des Lebenszyklus ist ein asynchroner Prozess, daher kann es einige Tage dauern, bis die Regel ausgeführt wird und der Bucket leer ist. Nachdem Amazon S3 die Regel zum ersten Mal ausführt, werden alle Objekte, die für den Ablauf in Frage kommen, zum Löschen markiert. Die Objekte, die zum Löschen markiert sind, werden Ihnen nicht mehr in Rechnung gestellt. Weitere Informationen finden Sie unter Wie leere ich einen Amazon-S3-Bucket mithilfe einer Lebenszykluskonfigurationsregel?.

Wenn Sie eine Lebenszykluskonfiguration verwenden, um Ihren Bucket zu leeren, sollte die Konfiguration aktuelle Versionen, nicht aktuelle Versionen, Löschmarkierungen und unvollständige mehrteilige Uploads enthalten.

Sie können Lebenszyklus-Konfigurationsregeln hinzufügen, sodass alle Objekte oder eine Untermenge davon mit einem spezifischen Schlüsselnamenpräfix ablaufen. Sie können z. B. eine Lebenszyklusregel so festlegen, dass Objekte einen Tag nach dem Erstellungsdatum ablaufen, um alle Objekte in einem Bucket zu entfernen.

Amazon S3 unterstützt eine Bucket-Lebenszyklusregel, mit der Sie unvollständige mehrteilige Uploads abbrechen können, die nicht innerhalb einer bestimmten Anzahl von Tagen nach der Initiierung abgeschlossen werden. Wir empfehlen, dass Sie diese Lebenszyklusregel konfigurieren, um Ihre Speicherkosten zu minimieren. Weitere Informationen finden Sie unter Konfigurieren einer Bucket-Lebenszykluskonfiguration zum Löschen unvollständiger mehrteiliger Uploads.

Weitere Informationen zur Verwendung einer Lebenszyklus-Konfiguration zum Leeren eines Buckets finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket und Auslaufende Objekte.

Leeren eines Buckets mit AWS CloudTrail konfiguriertem

AWS CloudTrail verfolgt Datenereignisse auf Objektebene in einem Amazon S3-Bucket, z. B. das Löschen von Objekten. Wenn Sie einen Bucket als Ziel verwenden, um Ihre CloudTrail Ereignisse zu protokollieren, und Objekte aus demselben Bucket löschen, erstellen Sie möglicherweise neue Objekte, während Sie Ihren Bucket leeren. Um dies zu verhindern, halten Sie Ihre AWS CloudTrail Trails an. Weitere Informationen dazu, wie Sie verhindern, dass Ihre CloudTrail Trails Ereignisse protokollieren, finden Sie unter Deaktivieren der Protokollierung für einen Trail im AWS CloudTrail - Benutzerhandbuch.

Eine weitere Alternative, um zu verhindern, dass CloudTrail Trails zum Bucket hinzugefügt werden, besteht darin, Ihrer Bucket-Richtlinie eine Deny-`s3:PutObject`Anweisung hinzuzufügen. Wenn Sie zu einem späteren Zeitpunkt neue Objekte im Bucket speichern möchten, müssen Sie diese Deny `s3:PutObject`-Anweisung entfernen. Weitere Informationen finden Sie unter Objektoperationen und IAM-JSON-Richtlinienelemente: Auswirkungen im IAM-Benutzerhandbuch.

Löschen eines Buckets

Sie können einen leeren Amazon-S3-Bucket löschen. Berücksichtigen Sie Folgendes, bevor Sie einen Bucket löschen:

- Bucket-Namen sind eindeutig. Wenn Sie einen Bucket löschen, kann ein anderer AWS Benutzer den Namen verwenden.
- Wenn der Bucket eine statische Website hostet und Sie eine gehostete Zone von Amazon Route 53, wie unter Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne beschrieben, erstellt und konfiguriert haben, müssen Sie die gehosteten Route 53-Zoneneinstellungen bereinigen, die sich auf den Bucket

beziehen. Weitere Informationen finden Sie unter Schritt 2: Löschen der von Route 53 gehosteten Zone.

- Wenn der Bucket Protokolldaten von Elastic Load Balancing (ELB) empfängt: Wir empfehlen, dass Sie die Übermittlung von ELB-Protokollen an den Bucket beenden, bevor Sie ihn löschen. Erstellt ein anderer Benutzer nach dem Löschen des Buckets einen Bucket mit demselben Namen, dann könnten Ihre Protokolldaten potenziell an diesen Bucket übermittelt werden. Informationen zu ELB-Zugriffsprotokollen finden Sie unter Zugriffsprotokollen im Benutzerhandbuch für Classic Load Balancers und Zugriffsprotokolle im Benutzerhandbuch für Application Load Balancers.

Fehlerbehebung

Wenn Sie einen Amazon-S3-Bucket nicht löschen können, sollten Sie Folgendes berücksichtigen:

- Stellen Sie sicher, dass der Bucket leer ist – Sie können nur Buckets löschen, bei denen keine Objekte enthalten sind. Stellen Sie sicher, dass der Bucket leer ist.
- Stellen Sie sicher, dass keine Zugriffspunkte angefügt sind – Sie können nur Buckets löschen, denen keine Zugriffspunkte angefügt sind. Löschen Sie alle Zugriffspunkte, die dem Bucket angefügt sind, bevor Sie den Bucket löschen.
- AWS Organizations Service-Kontrollrichtlinien (SCPs)– Eine Service-Kontrollrichtlinie kann die Löschberechtigung für einen Bucket verweigern. Informationen zu SCPs finden Sie unter Service-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- s3:DeleteBucket permissions – Wenn Sie einen Bucket nicht löschen können, wenden Sie sich an Ihren IAM-Administrator, um zu bestätigen, dass Sie über `-s3:DeleteBucket`Berechtigungen verfügen. Informationen zum Anzeigen oder Aktualisieren von IAM-Berechtigungen finden Sie unter Ändern von Berechtigungen für einen IAM-Benutzer im IAM-Benutzerhandbuch.
- s3:DeleteBucket deny statement – Wenn Sie über `s3:DeleteBucket` Berechtigungen in Ihrer IAM-Richtlinie verfügen und einen Bucket nicht löschen können, enthält die Bucket-Richtlinie möglicherweise eine Deny-Anweisung für `s3:DeleteBucket`. Buckets, die von erstellt wurden, ElasticBeanstalk verfügen standardmäßig über eine Richtlinie, die diese Anweisung enthält. Bevor Sie den Bucket löschen können, müssen Sie diese Anweisung oder die Bucket-Richtlinie löschen.

> **Important**
>
> Bucket-Namen sind eindeutig. Wenn Sie einen Bucket löschen, kann ein anderer AWS Benutzer den Namen verwenden. Wenn Sie weiterhin denselben Bucket-Namen verwenden

wollen, sollten Sie den Bucket nicht löschen. Wir empfehlen, den Bucket zu leeren und beizubehalten.

Verwenden der S3-Konsole

Einen S3-Bucket löschen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets die Option neben dem Namen des Buckets aus, den Sie löschen möchten, und wählen Sie dann oben auf der Seite Delete (Löschen) aus.
3. Bestätigen Sie auf der Seite Delete bucket (Bucket löschen), dass Sie den Bucket löschen möchten. Geben Sie dazu den Bucket-Namen in das Textfeld ein und wählen Sie Delete bucket (Bucket löschen).

   > **Note**
   >
   > Wenn der Bucket Objekte enthält, leeren Sie den Bucket, bevor Sie ihn löschen, indem Sie in der Fehlermeldung This bucket is not empty (Dieser Bucket ist nicht leer) den Link für eine leere Bucket-Konfiguration auswählen und die Anweisungen auf der Seite Empty bucket (Bucket leeren) befolgen. Kehren Sie dann zur Seite Delete bucket (Bucket löschen) zurück und löschen Sie den Bucket.

4. Wenn Sie überprüfen möchten, ob Sie den Bucket gelöscht haben, öffnen Sie die Buckets-Liste und geben den Namen des Buckets ein, den Sie gelöscht haben. Wenn der Buckets nicht gefunden wird, war der Löschvorgang erfolgreich.

Verwenden des AWS SDK for Java

Das folgende Beispiel zeigt, wie Sie einen Bucket mit dem AWS SDK for Java löschen. Zuerst löscht der Code Objekte im Bucket, und dann löscht er das Bucket. Weitere Informationen zu anderen AWS -SDKs finden Sie unter Tools für Amazon Web Services.

Java

Das folgende Java-Beispiel löscht einen Bucket mit Objekten. Zuerst löscht das Beispiel alle Objekte und dann den Bucket. Das Beispiel funktioniert für Buckets mit oder ohne aktiviertem Versioning.

> **Note**
>
> Bei Buckets ohne aktiviertes Versioning können Sie alle Objekte direkt löschen und danach den Bucket löschen. Bei Buckets mit aktiviertem Versioning müssen Sie zuerst alle Objektversionen löschen, bevor Sie den Bucket löschen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.Iterator;

public class DeleteBucket2 {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Delete all objects from the bucket. This is sufficient
            // for unversioned buckets. For versioned buckets, when you attempt to
 delete
```

```
            // objects, Amazon S3 inserts
            // delete markers for all objects, but doesn't delete the object
versions.
            // To delete objects from versioned buckets, delete all of the object
versions
            // before deleting
            // the bucket (see below for an example).
            ObjectListing objectListing = s3Client.listObjects(bucketName);
            while (true) {
                Iterator<S3ObjectSummary> objIter =
objectListing.getObjectSummaries().iterator();
                while (objIter.hasNext()) {
                    s3Client.deleteObject(bucketName, objIter.next().getKey());
                }

                // If the bucket contains many objects, the listObjects() call
                // might not return all of the objects in the first listing. Check
to
                // see whether the listing was truncated. If so, retrieve the next
page of
                // objects
                // and delete them.
                if (objectListing.isTruncated()) {
                    objectListing = s3Client.listNextBatchOfObjects(objectListing);
                } else {
                    break;
                }
            }

            // Delete all object versions (required for versioned buckets).
            VersionListing versionList = s3Client.listVersions(new
ListVersionsRequest().withBucketName(bucketName));
            while (true) {
                Iterator<S3VersionSummary> versionIter =
versionList.getVersionSummaries().iterator();
                while (versionIter.hasNext()) {
                    S3VersionSummary vs = versionIter.next();
                    s3Client.deleteVersion(bucketName, vs.getKey(),
vs.getVersionId());
                }

                if (versionList.isTruncated()) {
                    versionList = s3Client.listNextBatchOfVersions(versionList);
                } else {
```

```
                    break;
                }
            }

            // After all objects and object versions are deleted, delete the bucket.
            s3Client.deleteBucket(bucketName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
 couldn't
            // parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwenden der AWS CLI

Sie können einen Bucket löschen, der Objekte mit dem enthält AWS CLI , wenn das Versioning nicht aktiviert ist. Wenn Sie einen Bucket löschen, der Objekte enthält, werden alle Objekte in dem Bucket dauerhaft gelöscht. Dazu gehören auch Objekte, die in die `S3 Glacier`-Speicherklasse übergegangen sind.

Wenn für Ihren Bucket das Versioning nicht aktiviert ist, können Sie den AWS CLI Befehl `rb` (remove bucket) mit dem `--force` Parameter verwenden, um den Bucket und alle darin enthaltenen Objekte zu löschen. Dieser Befehl löscht zuerst alle Objekte und dann den Bucket.

Wenn das Versioning aktiviert ist, werden versionierte Objekte in diesem Prozess nicht gelöscht. Dadurch würde das Löschen des Buckets fehlschlagen, da der Bucket nicht leer wäre. Informationen zum Löschen versionierter Objekte finden Sie unter Löschen von Objektversionen.

```
$ aws s3 rb s3://bucket-name --force
```

Weitere Informationen finden Sie unter Verwenden von High-Level-S3-Befehlen mit im AWS Command Line Interface AWS Command Line Interface -Benutzerhandbuch.

Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert und Objekte werden automatisch unter Verwendung der serverseitigen Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt. Diese Verschlüsselungseinstellung gilt für alle Objekte in Ihren Amazon-S3-Buckets.

Wenn Sie mehr Kontrolle über Ihre Schlüssel benötigen, z. B. die Verwaltung der Schlüsselrotation und der Zugriffsrichtliniengewährungen, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) verwenden. Weitere Informationen zum Bearbeiten von KMS-Schlüsseln finden Sie unter Bearbeiten von Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service .

> **Note**
>
> Wir haben die Buckets geändert, sodass neue Objekt-Uploads automatisch verschlüsselt werden. Wenn Sie zuvor einen Bucket ohne Standardverschlüsselung erstellt haben, aktiviert Amazon S3 die Verschlüsselung für den Bucket standardmäßig mithilfe von SSE-S3. Die Standardverschlüsselungskonfiguration für einen vorhandenen Bucket, für den bereits SSE-S3 oder SSE-KMS konfiguriert ist, wird nicht geändert. Wenn Sie Ihre Objekte mit SSE-KMS verschlüsseln möchten, müssen Sie den Verschlüsselungstyp in Ihren Bucket-

Einstellungen ändern. Weitere Informationen finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

Wenn Sie Ihren Bucket für die Verwendung der Standardverschlüsselung mit SSE-KMS konfigurieren, können Sie auch S3-Bucket-Schlüssel aktivieren, um den Anforderungsverkehr von Amazon S3 zu zu zu verringern AWS KMS und die Verschlüsselungskosten zu senken. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

Um Buckets zu identifizieren, für die SSE-KMS für die Standardverschlüsselung aktiviert ist, können Sie Metriken von Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Weitere Informationen finden Sie unter Verwenden von S3 Storage Lens zum Schutz Ihrer Daten.

Wenn Sie die serverseitige Verschlüsselung verwenden, verschlüsselt Amazon S3 ein Objekt vor dem Speichern auf der Festplatte und entschlüsselt es beim Herunterladen des Objekts. Weitere Informationen zum Schutz von Daten mithilfe der serverseitigen Verschlüsselung und der Verwaltung der Verschlüsselungsschlüssel finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung.

Weitere Informationen zu den Berechtigungen, die für die Standardverschlüsselung erforderlich sind, finden Sie unter PutBucketEncryption in der API-Referenz zu Amazon Simple Storage Service.

Sie können das Amazon S3-Standardverschlüsselungsverhalten für einen S3-Bucket mithilfe der Amazon S3-Konsole, der - AWS SDKs, der Amazon S3-REST-API und der - AWS Befehlszeilenschnittstelle (AWS CLI) konfigurieren.

Verschlüsseln vorhandener Objekte

Zum Verschlüsseln Ihrer vorhandenen nicht verschlüsselten Amazon-S3-Objekte können Sie Amazon S3 Batch Operations verwenden. Sie stellen S3-Batch-Vorgänge eine Liste von Objekten bereit, für die Vorgänge ausgeführt werden sollen, und Batch-Vorgänge ruft die jeweilige API auf, um die angegebene Operation auszuführen. Mit der Operation Batch Operations Copy können Sie vorhandenen nicht verschlüsselte Objekte kopieren und die neuen verschlüsselten Objekte in denselben Bucket schreiben. Ein einzelner Batchoperations-Auftrag kann die angegebene Operation für Milliarden von Objekten ausführen. Weitere Informationen finden Sie unter Ausführung

umfangreicher Batch-Vorgänge für Amazon S3-Objekte durch. und im Beitrag des AWS Storage Blog Encrypting objects with Amazon S3 Batch Operations.

Sie können vorhandene Objekte auch mit der -`CopyObject`API-Operation oder dem `-copy-object` AWS CLI Befehl verschlüsseln. Weitere Informationen finden Sie unter AWS CLI und im Beitrag des AWS Storage Blog Encrypting existing objects with Amazon S3 Batch Operations.

> **Note**
>
> Amazon-S3-Buckets mit Standard-Bucket-Verschlüsselung, die auf SSE-KMS festgelegt ist, können nicht als Ziel-Buckets für the section called “Protokollierungs-Serverzugriff” verwendet werden. Für Zielbuckets des Server-Zugriffsprotokolls wird nur die Standard-Verschlüsselung SSE-S3 unterstützt.

Verwenden der SSE-KMS-Verschlüsselung für kontoübergreifende Vorgänge

Beachten Sie Folgendes, wenn Sie kontoübergreifende Operationen verschlüsseln:

- Wenn zum Zeitpunkt der Anforderung oder über die Standardverschlüsselungskonfiguration des Buckets kein AWS KMS key Amazon-Ressourcenname (ARN) oder Alias angegeben wird, wird die Von AWS verwalteter Schlüssel (`aws/s3`) verwendet.
- Wenn Sie S3-Objekte mithilfe von AWS Identity and Access Management (IAM)-Prinzipalen hochladen oder darauf zugreifen, die sich in derselben AWS-Konto wie Ihr KMS-Schlüssel befinden, können Sie die Von AWS verwalteter Schlüssel () verwenden`aws/s3`.
- Verwenden Sie einen vom Kunden verwalteten Schlüssel, wenn Sie kontoübergreifenden Zugriff auf Ihre S3-Objekte gewähren möchten. Sie können die Richtlinie eines vom Kunden verwalteten Schlüssel so konfigurieren, dass der Zugriff von einem anderen Konto aus möglich ist.
- Wenn Sie Ihren eigenen KMS-Schlüssel angeben, empfehlen wir, einen vollqualifizierten KMS-Schlüssel-ARN zu verwenden. Wenn Sie stattdessen einen KMS-Schlüsselalias verwenden, AWS KMS löst den Schlüssel innerhalb des Kontos des Anforderers auf. Dieses Verhalten kann dazu führen, dass Daten mit einem KMS-Schlüssel verschlüsselt werden, der dem Anforderer und nicht dem Bucket-Eigentümer gehört.
- Sie müssen einen Schlüssel angeben, für den Ihnen (dem Anforderer) die Berechtigung `Encrypt` erteilt wurde. Weitere Informationen finden Sie unter Schlüssel-Benutzern die Verwendung eines

KMS-Schlüssels für kryptografische Operationen gestatten im Entwicklerhandbuch zu AWS Key Management Service .

Weitere Informationen darüber, wann kundenverwaltete Schlüssel und - AWS verwaltete KMS-Schlüssel verwendet werden sollen, finden Sie unter Soll ich einen Von AWS verwalteter Schlüssel oder einen kundenverwalteten Schlüssel verwenden, um meine Objekte in Amazon S3 zu verschlüsseln?

Verwenden der Standard-Verschlüsselung mit der Replikation

Wenn Sie die Standard-Verschlüsselung für einen Replikations-Ziel-Bucket aktivieren, gilt das folgende Verschlüsselungsverhalten:

- Wenn Objekte im Quell-Bucket nicht verschlüsselt sind, werden die Replikatobjekte im Ziel-Bucket mithilfe der Einstellungen der Standard-Verschlüsselung des Ziel-Buckets verschlüsselt. Daher unterscheiden sich die ETags (Entity-Tags) der Quellobjekte von den ETags der Replikatobjekte. Wenn Sie Anwendungen haben, die ETags verwenden, müssen Sie diese Anwendungen aktualisieren, um diesen Unterschied auszugleichen.
- Wenn Objekte im Quell-Bucket mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3), serverseitiger Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder serverseitiger Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) verschlüsselt werden, verwenden die Replikatobjekte im Ziel-Bucket denselben Verschlüsselungstyp wie die Quellobjekte. Die Einstellungen der Standard-Verschlüsselung des Ziel-Buckets werden nicht verwendet.

Weitere Informationen über die Verwendung der Standard-Verschlüsselung mit SSE-KMS finden Sie unter Replizieren von verschlüsselten Objekten (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS).

Verwenden von Amazon S3-Bucket-Schlüsseln mit Standard-Verschlüsselung

Wenn Sie Ihren Bucket so konfigurieren, dass SSE-KMS als Standardverschlüsselungsverhalten für neue Objekte verwendet wird, können Sie auch S3-Bucket-Schlüssel konfigurieren. S3-Bucket-Schlüssel verringern die Anzahl der Transaktionen von Amazon S3 zu , AWS KMS um die Kosten für SSE-KMS zu senken.

Wenn Sie Ihren Bucket für die Verwendung von S3-Bucket-Schlüsseln für SSE-KMS bei neuen Objekten konfigurieren, AWS KMS generiert einen Schlüssel auf Bucket-Ebene, der verwendet wird, um einen eindeutigen Datenschlüssel für Objekte im Bucket zu erstellen. Dieser S3-Bucket-Schlüssel wird für einen zeitlich begrenzten Zeitraum in Amazon S3 verwendet, wodurch Amazon S3 keine Anforderungen an stellen muss AWS KMS , um Verschlüsselungsvorgänge abzuschließen.

Weitere Informationen zur Verwendung von S3-Bucket-Schlüsseln finden Sie unter Verwenden von Amazon-S3-Bucket-Schlüssel.

Konfigurieren der Standardverschlüsselung

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Für Amazon-S3-Buckets ist die Bucket-Verschlüsselung standardmäßig aktiviert und neue Objekte werden automatisch unter Verwendung der serverseitigen Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt. Diese Verschlüsselung gilt für alle neuen Objekte in Ihren Amazon-S3-Buckets und es fallen keine Kosten für Sie an.

Wenn Sie mehr Kontrolle über Ihre Verschlüsselungsschlüssel benötigen, z. B. die Verwaltung der Schlüsselrotation und der Zugriffsrichtliniengewährungen, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) verwenden. Weitere Informationen zu SSE-KMS finden Sie unter Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS). Weitere Informationen zu DSSE-KMS finden Sie unter the section called “Serverseitige Dual-Layer-Verschlüsselung (DSSE-KMS)”.

Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen. Weitere

Informationen zu kontoübergreifenden Berechtigungen für KMS-Schlüssel finden Sie unter Erstellen von KMS-Schlüsseln, die von anderen Konten verwendet werden können im Entwicklerhandbuch zu AWS Key Management Service .

Wenn Sie die Standard-Bucket-Verschlüsselung auf SSE-KMS festlegen, können Sie auch einen S3-Bucket-Schlüssel konfigurieren, um Ihre AWS KMS Anforderungskosten zu senken. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

> **Note**
>
> Wenn Sie verwendenPutBucketEncryption, um Ihre Standard-Bucket-Verschlüsselung auf SSE-KMS festzulegen, sollten Sie überprüfen, ob Ihre KMS-Schlüssel-ID korrekt ist. Amazon S3 validiert nicht die KMS-Schlüssel-ID, die in - PutBucketEncryption Anforderungen angegeben ist.

Es gibt keine zusätzlichen Gebühren für die Nutzung von Standard-Verschlüsselung für S3-Buckets. Für Anforderungen zum Konfigurieren des Standardverschlüsselungsverhaltens werden Standardgebühren für Amazon-S3-Anforderungen berechnet. Informationen zu Preisen finden Sie unter Amazon S3 – Preise. Für SSE-KMS und DSSE-KMS fallen AWS KMS Gebühren an, die unter – AWS KMS Preise aufgeführt sind.

Die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) wird für die Standardverschlüsselung nicht unterstützt.

Sie können die Amazon S3-Standardverschlüsselung für einen S3-Bucket mithilfe der Amazon S3-Konsole, der - AWS SDKs, der Amazon S3-REST-API und der AWS Command Line Interface (AWS CLI) konfigurieren.

Änderungen, die Sie vor dem Aktivieren der Standardverschlüsselung beachten sollten

Nachdem Sie die Standard-Verschlüsselung für einen Bucket aktiviert haben, gilt das folgende Verschlüsselungsverhalten:

- Es gibt keine Änderung der Verschlüsselung der Objekte, die vor der Aktivierung der Standard-Verschlüsselung im Bucket vorhanden waren.
- Wenn Sie Objekte nach der Aktivierung der Standard-Verschlüsselung hochladen:

  - Wenn Ihre PUT-Abfrage-Header keine Verschlüsselungsinformationen mit einschließen, verwendet Amazon S3 die Standardverschlüsselungseinstellungen des Buckets, um die Objekte zu verschlüsseln.
  - Wenn Ihre PUT-Anfrage-Header Verschlüsselungsinformationen mit einschließen, verwendet Amazon S3 die Verschlüsselungsinformationen der PUT-Anfrage, um Objekte zu verschlüsseln, bevor sie in Amazon S3 gespeichert werden.
- Wenn Sie die Option SSE-KMS oder DSSE-KMS für die Standardverschlüsselungskonfiguration verwenden, unterliegen Sie den Kontingenten der Anforderungen pro Sekunde (RPS) von AWS KMS. Weitere Informationen zu AWS KMS -Kontingenten und zum Anfordern einer Kontingenterhöhung finden Sie unter Kontingente im Entwicklerhandbuch zu AWS Key Management Service .

> **Note**
>
> Objekte, die hochgeladen wurden, bevor die Standardverschlüsselung aktiviert wurde, werden nicht verschlüsselt. Weitere Informationen zum Verschlüsseln vorhandener Objekte finden Sie unter the section called “Festlegen der Standard-Bucket-Verschlüsselung”.

Verwenden der S3-Konsole

Konfigurieren der Standardverschlüsselung für einen Amazon-S3-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des von Ihnen erstellten Buckets aus.
4. Wählen Sie die Registerkarte Eigenschaften aus.
5. Wählen Sie unter Default encryption (Standard-Verschlüsselung) Edit (Bearbeiten) aus.
6. Wählen Sie eine der folgenden Optionen unter Verschlüsselungstyp aus, um die Verschlüsselung zu konfigurieren:
   - Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)
   - Serverseitige Verschlüsselung mit - AWS Key Management Service Schlüsseln (SSE-KMS)

- Serverseitige Dual-Layer-Verschlüsselung mit - AWS Key Management Service Schlüsseln (DSSE-KMS)

> **Important**
>
> Wenn Sie die Optionen SSE-KMS oder DSSE-KMS für die Standardverschlüsselungskonfiguration verwenden, unterliegen Sie den Kontingenten der Anforderungen pro Sekunde (RPS) von AWS KMS. Weitere Informationen zu AWS KMS Kontingenten und zum Anfordern einer Kontingenterhöhung finden Sie unter Kontingente im AWS Key Management Service -Entwicklerhandbuch.

Buckets und neue Objekte werden standardmäßig mit SSE-S3 verschlüsselt, sofern Sie keine andere Art der Standardverschlüsselung für Ihre Buckets angeben. Weitere Informationen zur Standardverschlüsselung finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

Weitere Informationen zur Datenverschlüsselung mit der serverseitigen Amazon-S3-Verschlüsselung finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

7. Wenn Sie die serverseitige Verschlüsselung mit - AWS Key Management Service Schlüsseln (SSE-KMS) oder die serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service -Schlüsseln (DSSE-KMS) ausgewählt haben, gehen Sie wie folgt vor:

   a. Geben Sie unter AWS KMS -Schlüssel Ihren KMS-Schlüssel auf eine der folgenden Arten an:

      - Um aus einer Liste der verfügbaren KMS-Schlüssel auszuwählen, wählen Sie Aus Ihrem AWS KMS keys auswählen und wählen Sie Ihren KMS-Schlüssel aus der Liste der verfügbaren Schlüssel aus.

        Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen zu kundenverwalteten Schlüsseln finden Sie unter Kundenschlüssel und - AWS Schlüssel im AWS Key Management Service -Entwicklerhandbuch.

      - Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS key -ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.

- Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

  Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

> **Important**
>
> Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket aktiviert sind. Wenn Sie Choose from your KMS master keys (Auswahl aus Ihren KMS-Schlüsseln) auswählen, listet die S3-Konsole nur 100 KMS-Schlüssel pro Region auf. Wenn Sie über mehr als 100 KMS-Schlüssel in derselben Region verfügen, können Sie nur die ersten 100 KMS-Schlüssel in der S3-Konsole sehen. Um einen nicht in der Konsole aufgeführten KMS-Schlüssel zu verwenden, wählen Sie AWS KMS key -ARN eingeben aus und geben Sie den KMS-Schlüssel-ARN ein.
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung. Weitere Informationen zu diesen Schlüsseln finden Sie unter Symmetrische KMS-Verschlüsselungsschlüssel im Entwicklerhandbuch für AWS Key Management Service .

Weitere Informationen zur Verwendung von SSE-KMS mit Amazon S3 finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS). Weitere Informationen zur Verwendung von DSSE-KMS finden Sie unter the section called “Serverseitige Dual-Layer-Verschlüsselung (DSSE-KMS)”.

b. Wenn Sie Ihren Bucket für die Verwendung der Standardverschlüsselung mit SSE-KMS konfigurieren, können Sie auch einen S3-Bucket-Schlüssel aktivieren. S3-Bucket-Schlüssel senken die Verschlüsselungskosten, indem der Anforderungsverkehr von Amazon S3 zu verringert wird AWS KMS. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

   Um S3-Bucket-Schlüssel zu verwenden, wählen Sie unter Bucket Key (Bucket-Schlüssel) die Option Enable (Aktivieren).

> **Note**
>
> S3-Bucket-Schlüssel werden für DSSE-KMS nicht unterstützt.

8. Wählen Sie Änderungen speichern aus.

Verwenden der AWS CLI

Diese Beispiele zeigen Ihnen, wie Sie die Standardverschlüsselung mit SSE-S3 oder SSE-KMS mit einem S3-Bucket-Schlüssel konfigurieren.

Weitere Informationen zur Standardverschlüsselung finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets. Weitere Informationen zur Verwendung der AWS CLI zum Konfigurieren der Standardverschlüsselung finden Sie unter put-bucket-encryption.

Example – Standard-Verschlüsselung mit SSE-S3

In diesem Beispiel wird die Standardverschlüsselung von Buckets mit von Amazon S3 verwalteten Schlüsseln konfiguriert.

```
aws s3api put-bucket-encryption --bucket DOC-EXAMPLE-BUCKET --server-side-encryption-
configuration '{
    "Rules": [
        {
            "ApplyServerSideEncryptionByDefault": {
                "SSEAlgorithm": "AES256"
            }
        }
    ]
}'
```

Example – Standard-Verschlüsselung mit SSE-KMS mit einem S3-Bucket-Schlüssel

In diesem Beispiel wird die Standard-Bucket-Verschlüsselung mit SSE-KMS unter Verwendung eines S3-Bucket-Schlüssels konfiguriert.

```
aws s3api put-bucket-encryption --bucket DOC-EXAMPLE-BUCKET --server-side-encryption-
configuration '{
    "Rules": [
            {
```

```
                "ApplyServerSideEncryptionByDefault": {
                    "SSEAlgorithm": "aws:kms",
                    "KMSMasterKeyID": "KMS-Key-ARN"
                },
                "BucketKeyEnabled": true
            }
        ]
    }'
```

Verwenden der REST-API

Verwenden Sie die REST-API-Operation `PutBucketEncryption`, um die Standardverschlüsselung zu aktivieren und den Typ der serverseitigen Verschlüsselung festzulegen, der verwendet werden soll – SSE-S3, SSE-KMS oder DSSE-KMS.

Weitere Informationen finden Sie unter PutBucketEncryption in der API-Referenz zu Amazon Simple Storage Service.

Überwachung der Standardverschlüsselung mit AWS CloudTrail und Amazon EventBridge

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Sie können Konfigurationsanforderungen zur Standardverschlüsselung für Amazon-S3-Buckets mithilfe von AWS CloudTrail -Ereignissen verfolgen. Die folgenden API-Ereignisnamen werden in CloudTrail Protokollen verwendet:

- `PutBucketEncryption`

- `GetBucketEncryption`
- `DeleteBucketEncryption`

Sie können auch EventBridge Regeln erstellen, die den CloudTrail Ereignissen für diese API-Aufrufe entsprechen. Weitere Informationen zu CloudTrail Ereignissen finden Sie unter Aktivieren der Protokollierung für Objekte in einem Bucket mit der Konsole. Weitere Informationen zu EventBridge Ereignissen finden Sie unter Ereignisse von AWS-Services.

Sie können - CloudTrail Protokolle für Amazon S3-Aktionen auf Objektebene verwenden, um - PUT und -POSTAnforderungen an Amazon S3 zu verfolgen. Sie können diese Aktionen verwenden, um zu überprüfen, ob die Standardverschlüsselung zum Verschlüsseln von Objekten verwendet wird, wenn eingehende Anfragen PUT keine Verschlüsselungs-Header haben.

Wenn Amazon S3 ein Objekt mit den Einstellungen der Standardverschlüsselung verschlüsselt, enthält das Protokoll eins der folgenden Felder als Name-Wert-Paar: `"SSEApplied":"Default_SSE_S3"`, `"SSEApplied":"Default_SSE_KMS"` oder `"SSEApplied":"Default_DSSE_KMS"`.

Wenn Amazon S3 ein Objekt mit den PUT-Verschlüsselungs-Headern verschlüsselt, enthält das Protokoll eins der folgenden Felder als Name-Wert-Paar: `"SSEApplied":"SSE_S3"`, `"SSEApplied":"SSE_KMS"`, `"SSEApplied":"DSSE_KMS"` oder `"SSEApplied":"SSE_C"`.

Diese Informationen sind für mehrteilige Uploads in den Anforderungen der API-Operation `InitiateMultipartUpload` enthalten. Weitere Informationen zur Verwendung von CloudTrail und finden Sie CloudWatchunter Überwachen von Amazon S3.

Arbeiten mit Mountpoint für Amazon S3

Mountpoint für Amazon S3 ist ein Open-Source-Dateiclient mit hohem Durchsatz, um ein Amazon-S3-Bucket als lokales Dateisystem zu mounten. Mit Mountpoint können Ihre Anwendungen über Dateisystem-Operationen wie Öffnen und Lesen auf Objekte zugreifen, die in Amazon S3 gespeichert sind. Mountpoint übersetzt diese Operationen automatisch in API-Aufrufe für S3-Objekte, sodass Ihre Anwendungen über eine Dateischnittstelle auf den elastischen Speicher und den Durchsatz von Amazon S3 zugreifen können.

Mountpoint für Amazon S3 ist für Produktionszwecke in Ihren großen leseintensiven Anwendungen generell verfügbar: Data Lakes, Machine-Learning-Training, Bild-Rendering, Simulation autonomer Fahrzeuge, Extract, Transform, Load (ETL) und mehr.

Mountpoint unterstützt grundlegende Dateisystem-Operationen und kann Dateien mit einer Größe von bis zu 5 TB lesen. Diese Lösung kann vorhandene Dateien auflisten und lesen sowie neue erstellen. Sie kann keine vorhandenen Dateien ändern oder Verzeichnisse löschen und unterstützt keine symbolischen Links oder Dateisperren. Mountpoint ist ideal für Anwendungen geeignet, die nicht alle Features eines gemeinsam genutzten Dateisystems und Berechtigungen im POSIX-Stil, jedoch den elastischen Durchsatz von Amazon S3 zum Lesen und Schreiben großer S3-Datensätze benötigen. Einzelheiten finden Sie unter Mountpoint file system behavior auf GitHub. Für Workloads, die vollständige POSIX-Unterstützung erfordern, empfehlen wir Amazon FSx für Lustre und dessen Unterstützung für die Verknüpfung von S3-Buckets.

Mountpoint für Amazon S3 ist nur für Linux-Betriebssysteme verfügbar. Sie können Mountpoint für den Zugriff auf S3-Objekte in allen Speicherklassen außer S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive, S3 Intelligent-Tiering Archive Access Tier und S3 Intelligent-Tiering Deep Archive Access Tier verwenden.

Themen

Installieren von Mountpoint

Sie können vorgefertigte Pakete von Mountpoint für Amazon S3 über die Befehlszeile herunterladen und installieren. Die Anweisungen zum Herunterladen und Installieren von Mountpoint variieren je nach dem verwendeten Linux-Betriebssystem.

Themen

RPM-basierte Distributionen (Amazon Linux, Fedora, CentOS, RHEL)

1. Kopieren Sie die folgende Download-URL für Ihre Architektur.

   x86_64:

```
https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.rpm
```

ARM64 (Graviton):

```
https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.rpm
```

2. Laden Sie das Mountpoint für Amazon-S3-Paket herunter. Ersetzen Sie *download-link* durch die entsprechende Download-URL aus dem vorherigen Schritt.

```
wget download-link
```

3. (Optional) Überprüfen Sie die Authentizität und Integrität der heruntergeladenen Datei. Kopieren Sie zunächst die entsprechende Signatur-URL für Ihre Architektur.

   x86_64:

```
https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.rpm.asc
```

   ARM64 (Graviton):

```
https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.rpm.asc
```

   Sehen Sie sich als Nächstes Überprüfen der Signatur des Pakets für Mountpoint für Amazon S3 an.

4. Installieren Sie das Paket mit dem folgenden Befehl:

```
sudo yum install ./mount-s3.rpm
```

5. Stellen Sie sicher, dass Mountpoint erfolgreich installiert wurde, indem Sie den folgenden Befehl eingeben:

```
mount-s3 --version
```

   Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

```
mount-s3 1.0.0
```

DEB-basierte Distributionen (Debian, Ubuntu)

1. Kopieren Sie die Download-URL für Ihre Architektur.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.deb
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.deb
   ```

2. Laden Sie das Mountpoint für Amazon-S3-Paket herunter. Ersetzen Sie *download-link* durch die entsprechende Download-URL aus dem vorherigen Schritt.

   ```
   wget download-link
   ```

3. (Optional) Überprüfen Sie die Authentizität und Integrität der heruntergeladenen Datei. Kopieren Sie zunächst die Signatur-URL für Ihre Architektur.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.deb.asc
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.deb.asc
   ```

   Sehen Sie sich als Nächstes Überprüfen der Signatur des Pakets für Mountpoint für Amazon S3 an.

4. Installieren Sie das Paket mit dem folgenden Befehl:

   ```
   sudo apt-get install ./mount-s3.deb
   ```

5. Stellen Sie sicher, dass Mountpoint für Amazon S3 erfolgreich installiert wurde, indem Sie den folgenden Befehl ausführen:

   ```
   mount-s3 --version
   ```

Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

```
mount-s3 1.0.0
```

Andere Linux-Distributionen

1. Schlagen Sie in der Dokumentation Ihres Betriebssystems nach, um die erforderlichen Pakete FUSE und `libfuse2` zu installieren.
2. Kopieren Sie die Download-URL für Ihre Architektur.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.tar.gz
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.tar.gz
   ```
3. Laden Sie das Mountpoint für Amazon-S3-Paket herunter. Ersetzen Sie *`download-link`* durch die entsprechende Download-URL aus dem vorherigen Schritt.

   ```
   wget download-link
   ```
4. (Optional) Überprüfen Sie die Authentizität und Integrität der heruntergeladenen Datei. Kopieren Sie zunächst die Signatur-URL für Ihre Architektur.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.tar.gz.asc
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.tar.gz.asc
   ```

   Sehen Sie sich als Nächstes Überprüfen der Signatur des Pakets für Mountpoint für Amazon S3 an.
5. Installieren Sie das Paket mit dem folgenden Befehl:

```
sudo mkdir -p /opt/aws/mountpoint-s3 && sudo tar -C /opt/aws/mountpoint-s3 -xzf ./
mount-s3.tar.gz
```

6. Fügen Sie das `mount-s3`-Binary Ihrer `PATH`-Umgebungsvariablen hinzu. Fügen Sie in Ihrer `$HOME/.profile`-Datei die folgende Zeile an:

   ```
   export PATH=$PATH:/opt/aws/mountpoint-s3/bin
   ```

   Speichern Sie die `.profile`-Datei und führen Sie den folgenden Befehl aus:

   ```
   source $HOME/.profile
   ```

7. Stellen Sie sicher, dass Mountpoint für Amazon S3 erfolgreich installiert wurde, indem Sie den folgenden Befehl ausführen:

   ```
   mount-s3 --version
   ```

   Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

   ```
   mount-s3 1.0.0
   ```

Überprüfen der Signatur des Pakets für Mountpoint für Amazon S3

1. Installieren Sie GnuPG (der gpg-Befehl). Das ist erforderlich, um die Authentizität und Integrität eines heruntergeladenen Pakets für Mountpoint für Amazon S3 zu überprüfen. GnuPGist standardmäßig auf Amazon Linux Amazon Machine Images (AMIs) installiert. Fahren Sie nach der Installation von GnuPG mit Schritt 2 fort.
2. Laden Sie den öffentlichen Mountpoint-Schlüssel herunter, indem Sie den folgenden Befehl ausführen:

   ```
   wget https://s3.amazonaws.com/mountpoint-s3-release/public_keys/KEYS
   ```

3. Importieren Sie den öffentlichen Mountpoint-Schlüssel in Ihren Schlüsselbund, indem Sie den folgenden Befehl ausführen:

   ```
   gpg --import KEYS
   ```

4. Überprüfen Sie den Fingerabdruck des öffentlichen Mountpoint-Schlüssels, indem Sie den folgenden Befehl ausführen:

   ```
   gpg --fingerprint mountpoint-s3@amazon.com
   ```

   Vergewissern Sie sich, dass die angezeigte Fingerabdruck-Zeichenfolge mit der folgenden Zeichenfolge übereinstimmt:

   ```
   673F E406 1506 BB46 9A0E  F857 BE39 7A52 B086 DA5A
   ```

   Wenn die Zeichenfolge für den Fingerabdruck nicht übereinstimmt, stellen Sie die Installation von Mountpoint nicht fertig und wenden Sie sich an AWS Support.

5. Laden Sie die Paket-Signaturdatei herunter. Ersetzen Sie *signature-link* durch den entsprechenden Signaturlink aus den vorherigen Abschnitten.

   ```
   wget signature-link
   ```

6. Überprüfen Sie die Signatur des heruntergeladenen Pakets, indem Sie den folgenden Befehl ausführen. Ersetzen Sie *signature-filename* durch den Dateinamen aus dem vorherigen Schritt.

   ```
   gpg --verify signature-filename
   ```

   Führen Sie zum Beispiel bei RPM-basierten Distributionen (einschließlich Amazon Linux) den folgenden Befehl aus:

   ```
   gpg --verify mount-s3.rpm.asc
   ```

7. Die Ausgabe sollte den Text `Good signature` enthalten. Wenn die Ausgabe den Text `BAD signature` enthält, laden Sie die Mountpoint-Paketdatei erneut herunter und wiederholen Sie diese Schritte. Wenn das Problem weiterhin besteht, stellen Sie die Installation von Mountpoint nicht fertig und wenden Sie sich an AWS Support.

   Die Ausgabe kann eine Warnung über eine vertrauenswürdige Signatur enthalten. Dies deutet nicht auf ein Problem hin. Es bedeutet nur, dass Sie den öffentlichen Mountpoint-Schlüssel nicht unabhängig verifiziert haben.

Konfigurieren und Verwenden von Mountpoint

Um Mountpoint für Amazon S3 verwenden zu können, benötigt Ihr Host gültige AWS Anmeldeinformationen mit Zugriff auf den Bucket oder die Buckets, die Sie mounten möchten. Informationen zu verschiedenen Authentifizierungsmethoden finden Sie unter AWS - Anmeldedaten von Mountpoint auf GitHub.

Sie können beispielsweise zu diesem Zweck einen neuen AWS Identity and Access Management (IAM)-Benutzer und eine neue Rolle erstellen. Stellen Sie sicher, dass diese Rolle Zugriff auf den Bucket oder die Buckets hat, die Sie mounten möchten. Sie können die IAM-Rolle mit einem Instance-Profil an Ihre Amazon-EC2-Instance übergeben.

Verwenden von Mountpoint

Verwenden Sie Mountpoint für Amazon S3, um Folgendes zu tun:

1. Mounten Sie Buckets mit dem `mount-s3`-Befehl.

   Ersetzen Sie im folgenden Beispiel *DOC-EXAMPLE-BUCKET* durch den Namen Ihres S3-Buckets und ersetzen Sie *~/mnt* durch das Verzeichnis auf Ihrem Host, in dem Ihr S3-Bucket gemountet werden soll.

   ```
   mkdir ~/mnt
   mount-s3 DOC-EXAMPLE-BUCKET ~/mnt
   ```

   Da der Mountpoint-Client standardmäßig im Hintergrund ausgeführt wird, bietet Ihnen das *~/mnt*-Verzeichnis jetzt Zugriff auf die Objekte in Ihrem S3-Bucket.
2. Greifen Sie über Mountpoint auf die Objekte in Ihrem Bucket zu.

   Nachdem Sie Ihren Bucket lokal bereitgestellt haben, können Sie übliche Linux-Befehle, wie `cat` oder`ls`, verwenden, um mit Ihren S3-Objekten zu arbeiten. Mountpoint für Amazon S3 interpretiert Schlüssel in Ihrem S3-Bucket als Dateisystempfade. Dazu werden sie anhand des Schrägstrichs (`/`) getrennt. Wenn Sie zum Beispiel den Objektschlüssel `Data/2023-01-01.csv` in Ihrem Bucket haben, finden Sie ein Verzeichnis mit dem Namen `Data` in Ihrem Mountpoint-Dateisystem mit einer Datei namens `2023-01-01.csv` darin.

   Mountpoint für Amazon S3 implementiert die vollständige POSIX-Standardspezifikation für Dateisysteme absichtlich nicht. Mountpoint ist für Workloads optimiert, die über eine Dateisystemschnittstelle Lese- und Schreibzugriff mit hohem Durchsatz auf Daten benötigen,

die in Amazon S3 gespeichert aber ansonsten nicht auf Dateisystemfunktionen angewiesen sind. Weitere Informationen finden Sie unter Verhalten des Dateisystems von Mountpoint für Amazon S3 auf GitHub. Kunden, die eine umfassendere Dateisystemsemantik benötigen, sollten andere AWS Dateiservices wie Amazon Elastic File System (Amazon EFS) oder Amazon FSx in Betracht ziehen.

3. Heben Sie das Mounting Ihres Bucket mit dem `umount`-Befehl auf. Mit diesem Befehl wird Mounting Ihres S3-Bucket aufgehoben und Mountpoint wird beendet.

   Um den folgenden Beispielbefehl zu verwenden, ersetzen Sie *`~/mnt`* durch das Verzeichnis auf Ihrem Host, in dem Ihr S3-Bucket gemountet ist.

   ```
   umount ~/mnt
   ```

   > **Note**
   >
   > Wenn Sie eine Liste der Optionen für diesen Befehl erhalten möchten, führen Sie `umount --help` aus.

Weitere Einzelheiten zur Mountpoint-Konfiguration finden Sie unter S3-Bucket-Konfiguration und Konfiguration des Dateisystems auf GitHub.

Konfigurieren von Caching in Mountpoint

Sie können Mountpoint für Amazon S3 so konfigurieren, dass die zuletzt aus Ihren S3-Buckets im Amazon-EC2-Instance-Speicher oder in einem angefügten Amazon-EBS-Volume abgerufenen Daten im Cache gespeichert werden. Wenn Sie diese Daten im Cache speichern, kann dies die Leistung beschleunigen und die Kosten für wiederholte Datenzugriffe senken. Das Caching in Mountpoint ist ideal für Anwendungsfälle, in denen Sie wiederholt dieselben Daten lesen, die sich während der mehrfachen Lesevorgänge nicht ändern. Sie können das Caching beispielsweise bei Machine-Learning-Trainingsaufgaben verwenden, bei denen ein Trainingsdatensatz mehrmals gelesen werden muss, um die Modellgenauigkeit zu verbessern.

Wenn Sie einen S3-Bucket bereitstellen, können Sie optional das Caching mittels Flags aktivieren. Sie können den Speicherort und die Größe des Datencache sowie die Dauer der Beibehaltung von Metadaten im Cache konfigurieren. Wenn Sie einen Bucket bereitstellen und Caching aktiviert ist, erstellt Mountpoint ein leeres Unterverzeichnis am konfigurierten Cache-Speicherort, wenn

dieses Unterverzeichnis noch nicht vorhanden ist. Wenn Sie einen Bucket bereitstellen und diese Bereitstellung dann aufheben, löscht Mountpoint den Inhalt des Cache-Speicherorts. Weitere Informationen zum Konfigurieren und Verwenden von Caching in Mountpoint finden Sie unter Mountpoint für Amazon S3 Caching-Konfiguration auf GitHub.

Wenn Sie einen S3-Bucket bereitstellen, können Sie optional das Caching mittels des Flags `--cache CACHE_PATH` aktivieren. Ersetzen Sie im folgenden Beispiel *CACHE_PATH* durch den Dateipfad zu dem Verzeichnis, in dem Sie Ihre Daten zwischenspeichern möchten. Ersetzen Sie *DOC-EXAMPLE-BUCKET* durch den Namen Ihres S3-Buckets. Ersetzen Sie *~/mnt* durch das Verzeichnis auf Ihrem Host, in dem Ihr S3-Bucket bereitgestellt werden soll.

```
mkdir ~/mnt
mount-s3 --cache CACHE_PATH DOC-EXAMPLE-BUCKET ~/mnt
```

> **Important**
>
> Wenn Sie Caching aktivieren, speichert Mountpoint unverschlüsselte Objektinhalte aus Ihrem S3-Bucket an dem bei der Bereitstellung konfigurierten Caching-Speicherort. Zum Schutz Ihrer Daten sollten den Zugriff auf den Datencache-Speicherort einschränken.

Fehlerbehebung

Mountpoint für Amazon S3 wird von unterstützt AWS Support. Wenn Sie Hilfe benötigen, wenden Sie sich an das AWS Support -Zentrum.

Sie können Mountpoint-Probleme auch auf GitHub überprüfen und einreichen.

Wenn Sie in diesem Projekt ein potenzielles Sicherheitsproblem entdecken, bitten wir Sie über unsere Seite zur Meldung von Schwachstelle AWS Security darüber zu informieren. Schaffen Sie kein öffentliches GitHub-Problem.

Wenn sich Ihre Anwendung mit Mountpoint unerwartet verhält, können Sie Ihre Protokollinformationen überprüfen, um das Problem zu diagnostizieren.

Protokollierung

Standardmäßig sendet Mountpoint Protokollinformationen mit hohem Schweregrad an syslog.

Wenn Sie Protokolle auf den modernsten Linux-Distributionen, einschließlich Amazon Linux, anzeigen möchten, führen Sie den folgenden `journald`-Befehl aus:

```
journalctl -e SYSLOG_IDENTIFIER=mount-s3
```

Auf anderen Linux-Systemen werden `syslog`-Einträge wahrscheinlich in eine Datei wie `/var/log/syslog` geschrieben.

Sie können diese Protokolle verwenden, um Fehler in Ihrer Anwendung zu beheben. Wenn Ihre Anwendung beispielsweise versucht, eine vorhandene Datei zu überschreiben, schlägt der Vorgang fehl und im Protokoll wird eine Zeile angezeigt, die der folgenden ähnelt:

```
[WARN] open{req=12 ino=2}: mountpoint_s3::fuse: open failed: inode error: inode 2 (full
 key "README.md") is not writable
```

Weitere Informationen finden Sie unter Protokollierung in Mountpoint für Amazon S3 auf GitHub.

Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration

Amazon S3 Transfer Acceleration ist eine Funktion auf Bucket-Ebene, die die schnelle, einfache und sichere Übertragung von Dateien über große Entfernungen zwischen Ihrem Client und einem S3-Bucket ermöglicht. Transfer Acceleration wurde entwickelt, um Übertragungsgeschwindigkeiten aus der ganzen Welt in S3-Buckets zu optimieren. Transfer Acceleration nutzt die global verteilten Edge-Standorte in Amazon CloudFront. Sobald die Daten an einem Edge-Standort eingehen, werden sie über einen optimierten Netzwerkpfad an Ihren Amazon S3-Bucket weitergeleitet.

Wenn Sie Transfer Acceleration verwenden, fallen möglicherweise zusätzliche Gebühren für die Datenübertragung an. Weitere Informationen zu Preisen finden Sie unter Amazon S3-Preise.

Gründe für die Nutzung von Amazon S3 Transfer Acceleration

Es gibt verschiedene Gründe für die Verwendung von Transfer Acceleration auf einem Bucket:

- Ihre Kunden laden in einen zentralen Bucket aus der ganzen Welt hoch.
- Sie übertragen regelmäßig mehrere Gigabyte bis Terabyte von Daten über mehrere Kontinente hinweg.

- Sie können beim Hochladen in Amazon S3 nicht die gesamte verfügbare Bandbreite über das Internet nutzen.

Weitere Informationen darüber, wann Transfer Acceleration verwendet werden sollte, finden Sie in den FAQs zu Amazon S3.

Anforderungen für die Verwendung von Transfer Acceleration

Folgendes ist erforderlich, wenn Sie Transfer Acceleration für einen S3-Bucket verwenden:

- Transfer Acceleration wird nur bei virtuell gehosteten Anfragen unterstützt. Weitere Informationen zu virtuell gehosteten Anfragen finden Sie unter Senden von Anforderungen unter Verwendung der REST-API.
- Der Name des für Transfer Acceleration verwendeten Buckets muss DNS-konform sein, und er darf keine Punkte („.“) enthalten.
- Transfer Acceleration muss im Bucket aktiviert sein. Weitere Informationen finden Sie unter Aktivieren und Verwenden von S3 Transfer Acceleration.

  Nach der Aktivierung von Transfer Acceleration für einen Bucket kann es bis zu 20 Minuten dauern, bis sich die Datenübertragungsgeschwindigkeit in den Bucket erhöht.

  > **Note**
  >
  > Transfer Acceleration wird zurzeit für Buckets in den folgenden Regionen unterstützt:
  >
  > - Asien-Pazifik (Tokyo) (ap-northeast-1)
  > - Asien-Pazifik (Seoul): (ap-northeast-2)
  > - Asien-Pazifik (Mumbai): (ap-south-1)
  > - Asien-Pazifik (Singapur): (ap-southeast-1)
  > - Asien-Pazifik (Sydney): (ap-southeast-2)
  > - Kanada (Zentral): (ca-central-1)
  > - Europa (Frankfurt) (eu-central-1)
  > - Europa (Irland) (eu-west-1)
  > - Europa (London) (eu-west-2)
  > - Europa (Paris) (eu-west-3)
  > - Südamerika (São Paulo) (sa-east-1)

- USA Ost (Nord-Virginia): (us-east-1)
- USA Ost (Ohio): (us-east-2)
- USA West (Nordkalifornien) (us-west-1)
- USA West (Oregon): (us-west-2)

- Um auf den Bucket zuzugreifen, der für Transfer Acceleration konfiguriert ist, müssen Sie den Endpunkt *bucketname*`.s3-accelerate.amazonaws.com` verwenden. Oder den Dual-Stack-Endpunkt *bucketname*`.s3-accelerate.dualstack.amazonaws.com` für eine Verbindung mit dem aktivierten Bucket über IPv6 verwenden. Sie können weiterhin die regulären Endpunkte für die Standarddatenübertragung verwenden.
- Sie müssen der Bucket-Eigentümer sein, um den Transfer Acceleration-Status festlegen zu können. Der Bucket-Eigentümer kann anderen Benutzern Berechtigungen erteilen, um ihnen zu gestatten, den Beschleunigungsstatus für einen Bucket einzurichten. Die `s3:PutAccelerateConfiguration`-Berechtigung gestattet Benutzern, Transfer Acceleration für einen Bucket zu aktivieren oder zu deaktivieren. Die `-s3:GetAccelerateConfiguration`Berechtigung erlaubt Benutzern, den Transfer Acceleration-Status eines Buckets zurückzugeben, der entweder `Enabled` oder ist `Suspended`.

In den folgenden Abschnitten wird beschrieben, wie Sie beginnen und Amazon S3 Transfer Acceleration für die Übertragung von Daten verwenden.

Themen

Erste Schritte mit Amazon S3 Transfer Acceleration

Sie können Amazon S3 Transfer Acceleration für die schnelle, einfache und sichere Übertragung von Dateien über große Entfernungen zwischen Ihrem Client und einem S3-Bucket nutzen. Transfer Acceleration verwendet die global verteilten Edge-Standorte in Amazon CloudFront. Sobald die Daten an einem Edge-Standort eingehen, werden sie über einen optimierten Netzwerkpfad an Ihren Amazon S3-Bucket weitergeleitet.

Um mit der Verwendung von Amazon S3 Transfer Acceleration zu beginnen, führen Sie die folgenden Schritte aus:

1. Aktivieren der Transfer Acceleration auf einem Bucket

   Sie können Transfer Acceleration für einen Bucket auf eine der folgenden Arten aktivieren:

   - Verwenden der Amazon S3-Konsole
   - Verwenden der REST API PUT Bucket accelerate-Operation.
   - Verwenden Sie die AWS CLI und AWS SDKs . Weitere Informationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

   Weitere Informationen finden Sie unter Aktivieren und Verwenden von S3 Transfer Acceleration.

   > **Note**
   >
   > Damit Ihr Bucket mit Transfer Acceleration arbeiten kann, muss sein Name den DNS-Benennungsanforderungen entsprechen und darf keine Punkte (".") enthalten.

2. Übertragen von Daten in den und aus dem beschleunigungsfähigen Bucket

   Verwenden Sie einen der folgenden s3-beschleunigten Endpunkt-Domänennamen:

   - Um auf einen beschleunigungsfähigen Bucket zuzugreifen, nutzen Sie *`bucketname`*`.s3-accelerate.amazonaws.com`.
   - Um auf einen beschleunigungsfähigen Bucket über IPv6 zuzugreifen, nutzen Sie *`bucketname`*`.s3-accelerate.dualstack.amazonaws.com`.

     Amazon S3-Dual-Stack-Endpunkte unterstützen Anforderungen an S3-Buckets über IPv6 und IPv4. Der Transfer Acceleration-Dual-Stack-Endpunkt verwendet nur den virtuellen Hosting-Endpunktnamen. Weitere Informationen finden Sie unter Erste Schritte für Anforderungen über IPv6 und Verwenden von Amazon-S3-Dual-Stack-Endpunkten.

   > **Note**
   >
   > Ihre Anwendung zur Datenübertragung muss eine der beiden folgenden Endpunkt-Typen verwenden, um auf den Bucket zugreifen und die Datenübertragung beschleunigen zu können: `.s3-accelerate.amazonaws.com` oder `.s3-accelerate.dualstack.amazonaws.com` für den Dual-Stack-Endpunkt. Wenn Sie

die Standarddatenübertragung verwenden möchten, können Sie weiterhin die regulären Endpunkte nutzen.

Sie können in Ihren Amazon S3 PUT object- und GET object-Anfragen auf den `s3-accelerate`-Endpunkt-Domänennamen verweisen, nachdem Sie Amazon S3 Transfer Acceleration aktiviert haben. Angenommen, Sie haben derzeit eine REST-API-Anwendung, die PUT Object verwendet, die den Hostnamen `mybucket.s3.us-east-1.amazonaws.com` in der Anfrage PUT verwendet. Um PUT zu beschleunigen, ändern Sie den Hostnamen in Ihrer Anfrage in `mybucket.s3-accelerate.amazonaws.com`. Um wieder die Standard-Upload-Geschwindigkeit zu verwenden, ändern Sie den Namen zurück in `mybucket.s3.us-east-1.amazonaws.com`.

Nachdem Transfer Acceleration aktiviert wurde, kann es bis zu 20 Minuten dauern, bis Sie den Leistungsvorteil wahrnehmen. Der beschleunigte Endpunkt steht zur Verfügung, sobald Sie Transfer Acceleration aktivieren.

Sie können den beschleunigten Endpunkt in der AWS CLI, AWS SDKs und anderen Tools verwenden, die Daten zu und von Amazon S3 übertragen. Wenn Sie die AWS SDKs verwenden, verwenden einige der unterstützten Sprachen ein Flag für eine Client-Konfiguration mit beschleunigtem Endpunkt, sodass Sie den Endpunkt für Transfer Acceleration nicht explizit auf setzen müssen*bucketname*`.s3-accelerate.amazonaws.com`. Beispiele für das Flag für eine Client-Konfiguration mit beschleunigtem Endpunkt finden Sie unter Aktivieren und Verwenden von S3 Transfer Acceleration.

Sie können alle Amazon S3-Vorgänge über die Endpunkte der Transfer Acceleration mit Ausnahme der Folgenden verwenden:

- GET Service (Buckets auflisten)
- PUT Bucket (Bucket erstellen)
- DELETE Bucket

Außerdem unterstützt Amazon S3 Transfer Acceleration keine regionsübergreifenden Kopien mit PUT Object – Copy.

Aktivieren und Verwenden von S3 Transfer Acceleration

Sie können Amazon S3 Transfer Acceleration verwenden, um Dateien schnell und sicher über große Entfernungen zwischen Ihrem Kunden und einem S3-Bucket zu übertragen. Sie können Transfer Acceleration über die S3-Konsole, die AWS Command Line Interface (AWS CLI), die API oder die AWS SDKs aktivieren.

Dieser Abschnitt bietet Beispiele für die Aktivierung von Amazon S3 Transfer Acceleration und die Verwendung des beschleunigten Endpunkts für den aktivierten Bucket.

Weitere Informationen zu den Anforderungen für Transfer Acceleration finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.

Verwenden der S3-Konsole

> **Note**
>
> Wenn Sie beschleunigte und nicht beschleunigte Upload-Geschwindigkeiten vergleichen möchten, öffnen Sie das Amazon S3 Transfer Acceleration Speed Comparison-Tool.
> Das Speed Comparison-Tool verwendet mehrteilige Uploads, um eine Datei von Ihrem Browser in verschiedene AWS-Regionen mit und ohne Amazon S3 Transfer Acceleration zu übertragen. Sie können die Upload-Geschwindigkeit für direkte Uploads vergleichen und beschleunigte Uploads nach Region übertragen.

Die Transfer Acceleration für einen S3-Bucket aktivieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie Transfer Acceleration aktivieren möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Transfer acceleration (Beschleunigung übertragen) die Option Edit (Bearbeiten) aus.
5. Wählen Sie Enable (Aktivieren) und wählen Sie Save changes (Änderungen speichern) aus.

Zugriff auf beschleunigte Datenübertragungen

1. Nachdem Amazon S3 die Übertragungsbeschleunigung für Ihren Bucket aktiviert hat, sehen Sie sich den Tab Properties (Eigenschaften) für den Bucket an.
2. Unter Transfer acceleration (Übertragungsbeschleunigung) wird unter Accelerated endpoint (beschleunigter Endpunkt) der Endpunkt der Übertragungsbeschleunigung für Ihren Bucket angezeigt. Verwenden Sie diesen Endpunkt, um auf beschleunigte Datenübertragungen von und zu Ihrem Bucket zuzugreifen.

   Wenn Sie die Transfer Acceleration aussetzen, funktioniert der beschleunigte Endpunkt nicht mehr.

Verwenden der AWS CLI

Im Folgenden finden Sie Beispiele für AWS CLI Befehle, die für Transfer Acceleration verwendet werden. Anweisungen zum Einrichten der finden Sie AWS CLI unter Entwickeln mit Amazon S3 über die AWS CLI.

Aktivieren der Transfer Acceleration auf einem Bucket

Verwenden Sie den AWS CLI put-bucket-accelerate-configuration Befehl , um Transfer Acceleration für einen Bucket zu aktivieren oder auszusetzen.

Im folgenden Beispiel wird `Status=Enabled` zur Aktivierung von Transfer Acceleration für einen Bucket festgelegt. Sie verwenden `Status=Suspended`, um Transfer Acceleration auszusetzen.

Example

```
$ aws s3api put-bucket-accelerate-configuration --bucket bucketname --accelerate-
configuration Status=Enabled
```

Nutzen der Transfer Acceleration

Sie können alle Amazon S3-Anfragen von s3- und s3api- AWS CLI Befehlen an den beschleunigten Endpunkt weiterleiten: `s3-accelerate.amazonaws.com`. Legen Sie dazu den Konfigurationswert `true` in einem Profil in Ihrer AWS Config-Datei `use_accelerate_endpoint` auf fest. Transfer Acceleration muss für Ihren Bucket aktiviert sein, um den beschleunigten Endpunkt nutzen zu können.

Alle Anforderungen werden mit der virtuellen Bucket-Adressierung gesendet: `my-bucket.s3-accelerate.amazonaws.com`. `ListBuckets`-, `CreateBucket`-, und `DeleteBucket`-Anfragen werden nicht an den beschleunigten Endpunkt gesendet, da der Endpunkt diese Vorgänge nicht unterstützt.

Weitere Informationen zu `use_accelerate_endpoint` finden Sie in der AWS CLI -S3-Konfiguration in der AWS CLI -Befehlsreferenz.

Das folgende Beispiel setzt `use_accelerate_endpoint` im Standardprofil auf `true`.

Example

```
$ aws configure set default.s3.use_accelerate_endpoint true
```

Wenn Sie den beschleunigten Endpunkt für einige AWS CLI Befehle, aber nicht für andere verwenden möchten, können Sie eine der beiden folgenden Methoden verwenden:

- Verwenden Sie den beschleunigten Endpunkt für jeden s3- oder s3api-Befehl, indem Sie den Parameter `--endpoint-url` auf `https://s3-accelerate.amazonaws.com` setzen.
- Richten Sie separate Profile in Ihrer AWS Config-Datei ein. Legen Sie beispielsweise ein Profil an, das `use_accelerate_endpoint` auf `true` setzt, und ein Profil, das `use_accelerate_endpoint` nicht setzt. Wenn Sie einen Befehl ausführen, geben Sie an, welches Profil Sie verwenden möchten, abhängig davon, ob Sie den beschleunigten Endpunkt verwenden möchten.

Hochladen eines Objekts in einen Bucket, der für Transfer Acceleration geeignet ist

Das folgende Beispiel lädt eine Datei in einen Bucket hoch, der für Transfer Acceleration konfiguriert ist. Dazu verwendet es das Standardprofil, das für die Verwendung des beschleunigten Endpunkts konfiguriert wurde.

Example

```
$ aws s3 cp file.txt s3://bucketname/keyname --region region
```

Das folgende Beispiel lädt eine Datei in einen Bucket hoch, der für Transfer Acceleration konfiguriert ist. Dazu verwendet es den Parameter `--endpoint-url` für die Angabe des beschleunigten Endpunkts.

Example

```
$ aws configure set s3.addressing_style virtual
$ aws s3 cp file.txt s3://bucketname/keyname --region region --endpoint-url https://s3-accelerate.amazonaws.com
```

Verwenden der AWS SDKs

Im Folgenden finden Sie Beispiele für die Verwendung von Transfer Acceleration zum Hochladen von Objekten in Amazon S3 mit dem AWS SDK. Einige der von AWS SDK unterstützten Sprachen (z. B. Java und .NET) verwenden ein Flag für eine Client-Konfiguration mit beschleunigtem Endpunkt, sodass Sie den Endpunkt für Transfer Acceleration nicht explizit auf den *Bucket-Namen* .s3-accelerate.amazonaws.com setzen müssen.

Java

Example

Das folgende Beispiel zeigt, wie Sie einen beschleunigten Endpunkt für das Hochladen eines Objekts in Amazon S3 verwenden. Das Beispiel erledigt Folgendes:

- Erstellt einen `AmazonS3Client`, der für die Verwendung von beschleunigten Endpunkten konfiguriert ist. Für alle Buckets, auf die der Client zugreift, muss Transfer Acceleration aktiviert sein.
- Aktiviert Transfer Acceleration auf einem angegebenen Bucket. Dieser Schritt ist nur erforderlich, wenn für den von Ihnen angegebenen Bucket nicht bereits Transfer Acceleration aktiviert ist.
- Überprüft, ob Transfer Acceleration für den angegebenen Bucket aktiviert ist.
- Lädt ein neues Objekt in den angegebenen Bucket hoch und verwendet dazu den beschleunigten Endpunkt des Buckets.

Weitere Informationen zur Verwendung von Transfer Acceleration finden Sie unter Erste Schritte mit Amazon S3 Transfer Acceleration. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
```

```
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketAccelerateConfiguration;
import com.amazonaws.services.s3.model.BucketAccelerateStatus;
import com.amazonaws.services.s3.model.GetBucketAccelerateConfigurationRequest;
import com.amazonaws.services.s3.model.SetBucketAccelerateConfigurationRequest;

public class TransferAcceleration {
    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";

        try {
            // Create an Amazon S3 client that is configured to use the accelerate
 endpoint.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .enableAccelerateMode()
                    .build();

            // Enable Transfer Acceleration for the specified bucket.
            s3Client.setBucketAccelerateConfiguration(
                    new SetBucketAccelerateConfigurationRequest(bucketName,
                            new BucketAccelerateConfiguration(
                                    BucketAccelerateStatus.Enabled)));

            // Verify that transfer acceleration is enabled for the bucket.
            String accelerateStatus = s3Client.getBucketAccelerateConfiguration(
                    new GetBucketAccelerateConfigurationRequest(bucketName))
                    .getStatus();
            System.out.println("Bucket accelerate status: " + accelerateStatus);

            // Upload a new object using the accelerate endpoint.
            s3Client.putObject(bucketName, keyName, "Test object for transfer
 acceleration");
            System.out.println("Object \"" + keyName + "\" uploaded with transfer
 acceleration.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
```

```
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Das folgende Beispiel zeigt, wie Sie die verwenden, AWS SDK for .NET um Transfer Acceleration für einen Bucket zu aktivieren. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TransferAccelerationTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            EnableAccelerationAsync().Wait();
        }

        static async Task EnableAccelerationAsync()
        {
                try
                {
```

```
                    var putRequest = new PutBucketAccelerateConfigurationRequest
                    {
                        BucketName = bucketName,
                        AccelerateConfiguration = new AccelerateConfiguration
                        {
                            Status = BucketAccelerateStatus.Enabled
                        }
                    };
                    await
 s3Client.PutBucketAccelerateConfigurationAsync(putRequest);

                    var getRequest = new GetBucketAccelerateConfigurationRequest
                    {
                        BucketName = bucketName
                    };
                    var response = await
 s3Client.GetBucketAccelerateConfigurationAsync(getRequest);

                    Console.WriteLine("Acceleration state = '{0}' ",
 response.Status);
                }
                catch (AmazonS3Exception amazonS3Exception)
                {
                    Console.WriteLine(
                        "Error occurred. Message:'{0}' when setting transfer
 acceleration",
                        amazonS3Exception.Message);
                }
        }
    }
}
```

Wenn Sie ein Objekt in einen Bucket hochladen, für den Transfer Acceleration aktiviert ist, verwenden Sie für die Angabe des beschleunigten Endpunkts den Zeitpunkt, an dem ein Client erstellt wurde.

```
var client = new AmazonS3Client(new AmazonS3Config
            {
                RegionEndpoint = TestRegionEndpoint,
                UseAccelerateEndpoint = true
            }
```

Javascript

Ein Beispiel für die Aktivierung von Transfer Acceleration mithilfe des AWS -SDK für JavaScriptfinden Sie unter Aufrufen des putBucketAccelerateKonfigurationsvorgangs in der AWS -SDK für JavaScript API-Referenz.

Python (Boto)

Ein Beispiel für die Aktivierung von Transfer Acceleration mithilfe des SDK for Python finden Sie unter put_bucket_accelerate_configuration in der API–Referenz für AWS -SDK für Python (Boto3).

Other

Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispielcode und Bibliotheken.

Verwenden der REST-API

Verwenden Sie die REST-API-Operation `PutBucketAccelerateConfiguration`, um die beschleunigte Konfiguration für einen vorhandenen Bucket zu aktivieren.

Weitere Informationen finden Sie unter PutBucketAccelerateConfiguration in der API-Referenz zu Amazon Simple Storage Service.

Verwenden des Amazon S3 Transfer Acceleration Speed Comparison-Tools

Sie können das Amazon S3 Transfer Acceleration Speed Comparison Tool verwenden, um beschleunigte und nicht beschleunigte Upload-Geschwindigkeiten in allen Amazon S3-Regionen zu vergleichen. Das Speed Comparison-Tool verwendet mehrteilige Uploads, um eine Datei von Ihrem Browser in verschiedene Amazon S3-Regionen mit und ohne die Verwendung von Transfer Acceleration zu übertragen.

Sie können mit einer der folgenden Methoden auf das Speed Comparison-Tool zugreifen:

- Kopieren Sie die folgende URL in Ihr Browserfenster und ersetzen Sie *`region`* durch die von AWS-Region Ihnen verwendete (z. B. `us-west-2`) und *`yourBucketName`* durch den Namen des auszuwertenden Buckets:

```
https://s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/
accelerate-speed-comparsion.html?
region=region&origBucketName=yourBucketName
```

Eine Liste der von Amazon S3 unterstützten Regionen finden Sie unter Endpunkte und Kontingente von Amazon S3 in der Allgemeine AWS-Referenz.

- Verwenden der Amazon S3-Konsole

Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung

Im Allgemeinen zahlen Bucket-Eigentümer alle Amazon-S3-Speicher- und -Datenübertragungskosten, die ihrem Bucket zuzuordnen sind. Sie können jedoch einen Bucket als Bucket mit Zahlung durch den Anforderer konfigurieren. Bei Buckets mit Zahlung durch den Anforderer zahlt der Auftraggeber statt des Bucket-Eigentümers die Kosten für die Anforderung und den Daten-Download aus dem Bucket. Der Bucket-Eigentümer zahlt immer die Kosten für das Speichern der Daten.

In der Regel konfigurieren Sie Buckets dann als Buckets mit Zahlung durch den Anforderer, wenn Sie Daten freigeben möchten, aber keine Gebühren dafür übernehmen wollen, wenn andere auf die Daten zugreifen. Sie können beispielsweise Buckets mit Zahlung durch den Anforderer verwenden, wenn Sie große Datasets bereitstellen, beispielsweise Postleitzahlenverzeichnisse, Referenzdaten, Geodateninformationen oder Web-Crawling-Daten.

> **Important**
>
> Wenn Sie die Zahlung durch den Anforderer für einen Bucket aktivieren, ist kein anonymer Zugriff auf den Bucket zulässig.

Sie müssen alle Anforderungen für Buckets mit Zahlung durch den Anforderer authentifizieren. Die Anfrageauthentifizierung ermöglicht Amazon S3, den Auftraggeber zu identifizieren und ihm seine Verwendung des Buckets mit Zahlung durch den Anforderer in Rechnung zu stellen.

Wenn der Anforderer vor der Anforderung eine AWS Identity and Access Management (IAM)-Rolle annimmt, wird das Konto, zu dem die Rolle gehört, für die Anforderung belastet. Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

Nachdem Sie einen Bucket als Bucket mit Zahlung durch den Anforderer konfiguriert haben, müssen die Anforderer ihnen nachweisen, dass ihnen die Gebühren für die Anforderung und den Datendownload in Rechnung gestellt werden. Um zu zeigen, dass sie die Gebühren akzeptieren, müssen Anforderer entweder `x-amz-request-payer` als Header in ihre API-Anfrage für DELETE-, GET-, HEAD-, POST- und PUT-Anfragen aufnehmen oder den `RequestPayer` Parameter in ihre REST-Anfrage aufnehmen. Bei CLI-Anforderungen können Anforderer den `--request-payer` Parameter verwenden.

Example – Verwenden von Zahlung durch den Anforderer beim Löschen eines Objekts

Um das folgende DeleteObjectVersion API-Beispiel zu verwenden, ersetzen Sie durch *`user input placeholders`* Ihre eigenen Informationen.

```
DELETE /Key+?versionId=VersionId HTTP/1.1
Host: Bucket.s3.amazonaws.com
x-amz-mfa: MFA
x-amz-request-payer: RequestPayer
x-amz-bypass-governance-retention: BypassGovernanceRetention
x-amz-expected-bucket-owner: ExpectedBucketOwner
```

Wenn der Anforderer Objekte mithilfe der RestoreObject-API wiederherstellt, wird die Zahlung durch den Anforderer unterstützt, solange der -`x-amz-request-payer`Header oder der -`RequestPayer`Parameter in der Anforderung enthalten sind. Der Anforderer zahlt jedoch nur die Kosten der Anforderung. Der Bucket-Eigentümer zahlt die Abrufgebühren.

Buckets mit Zahlung durch den Anforderer unterstützen die folgenden Funktionen nicht:

- Anonyme Anforderungen
- SOAP-Anforderungen
- Die Verwendung eines Buckets mit Zahlung durch den Anforderer als Ziel-Bucket für die Endbenutzer-Protokollierung, oder umgekehrt. Sie können jedoch die Endbenutzer-Protokollierung für einen Bucket mit Zahlung durch den Anforderer aktivieren, wenn es sich beim Ziel-Bucket nicht um einen Bucket mit Zahlung durch den Anforderer handelt.

Die Gebühren bei Zahlung durch den Anforderer

Die Verteilung der Gebühren für erfolgreiche Anforderungen mit Zahlung durch den Anforderer ist ganz einfach: Der Anforderer zahlt für die Datenübertragung und die Anforderung, der Bucket-

Eigentümer für die Speicherung der Daten. Dem Bucket-Eigentümer werden jedoch nur unter den folgenden Bedingungen Gebühren für die Anforderung in Rechnung gestellt:

- Der Auftraggeber gibt den Parameter `x-amz-request-payer` nicht im Header (DELETE, GET, HEAD, POST und PUT) oder als Parameter (REST) in der Anforderung (HTTP-Code 403) an.
- Die Authentifizierung der Anforderung schlägt fehl (HTTP-Code 403).
- die Anforderung ist anonym (HTTP-Code 403).
- die Anforderung ist eine SOAP-Anforderung.

Weitere Informationen zur Zahlung durch den Anforderer finden Sie in den folgenden Themen.

Themen

Konfigurieren von Zahlung durch den Anforderer für einen Bucket

Sie können einen Amazon-S3-Bucket als Bucket mit Zahlung durch den Anforderer konfigurieren, damit anstelle des Bucket-Eigentümers der Anforderer die Kosten für die Anforderung und den Datendownload übernimmt.

Dieser Abschnitt enthält Beispiele für die Konfiguration von Zahlung durch den Anforderer auf einem Amazon-S3-Bucket über die Konsole und die REST-API.

Verwenden der S3-Konsole

So aktivieren Sie die Zahlung durch den Anforderer für einen S3-Bucket:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie die Zahlung durch den Anforderer aktivieren möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Requester pays (Zahlung durch Anforderer) die Option Edit (Bearbeiten).

5. Wählen Sie Enable (Aktivieren) und wählen Sie Save changes (Änderungen speichern) aus.

   Amazon S3 aktiviert die Zahlung durch den Anforderer für Ihren Bucket und zeigt Ihre Bucket-Übersichtan. Unter Zahlung durch den Anforderer sehen Sie Aktiviert

Verwenden der REST-API

Nur der Bucket-Eigentümer kann den Konfigurationswert `RequestPaymentConfiguration.payer` eines Buckets auf `BucketOwner` (den Standardwert) oder `Requester` setzen. Die Einrichtung der Ressource `requestPayment` ist optional. Standardmäßig ist der Bucket kein Bucket mit Zahlung durch den Anforderer.

Um einen Bucket mit Zahlung durch den Anforderer in einen regulären Bucket umzuwandeln, verwenden Sie den Wert `BucketOwner`. In der Regel verwenden Sie `BucketOwner` zum Hochladen von Daten in den Amazon-S3-Bucket. Dann würden Sie den Wert auf `Requester` setzen, bevor Sie die Objekte in dem Bucket veröffentlichen.

Einrichten von requestPayment

- Verwenden Sie eine PUT-Anforderung, um den Wert `Payer` für einen bestimmten Bucket auf `Requester` zu setzen.

```
PUT ?requestPayment HTTP/1.1
Host: [BucketName].s3.amazonaws.com
Content-Length: 173
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]

<RequestPaymentConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Payer>Requester</Payer>
</RequestPaymentConfiguration>
```

War die Anfrage erfolgreich, gibt Amazon S3 eine Antwort zurück, die etwa wie folgt aussieht.

```
HTTP/1.1 200 OK
x-amz-id-2: [id]
x-amz-request-id: [request_id]
Date: Wed, 01 Mar 2009 12:00:00 GMT
Content-Length: 0
Connection: close
```

```
Server: AmazonS3
x-amz-request-charged:requester
```

Sie können Zahlung durch den Anforderer nur auf Bucket-Ebene festlegen. Sie können Zahlung durch den Anforderer nicht für bestimmte Objekte innerhalb des Buckets festlegen.

Sie können einen Bucket jederzeit als `BucketOwner` oder `Requester` konfigurieren. Es kann jedoch einige Minuten dauern, bis der neue Konfigurationswert aktiv ist.

> **Note**
>
> Bucket-Eigentümer, die vorsignierte URLs ausgeben, sollten sich gut überlegen, ob sie einen Bucket als Bucket mit Zahlung durch den Anforderer konfigurieren sollten, insbesondere, wenn die URL eine lange Lebensdauer hat. Der Bucket-Eigentümer muss jedes Mal die Gebühren zahlen, wenn der Auftraggeber eine vorsignierte URL verwendet, die die Anmeldeinformationen des Bucket-Eigentümers verwendet.

Abrufen der requestPayment-Konfiguration mit der REST-API

Sie können den `Payer`-Wert ermitteln, der für einen Bucket eingerichtet ist, indem Sie die Ressource `requestPayment` abrufen.

Rückkehr zur requestPayment-Ressource

- Verwenden Sie eine GET-Anforderung, um die `requestPayment`-Ressource zu erhalten, wie in der folgenden Anforderung gezeigt.

```
GET ?requestPayment HTTP/1.1
Host: [BucketName].s3.amazonaws.com
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]
```

War die Anfrage erfolgreich, gibt Amazon S3 eine Antwort zurück, die etwa wie folgt aussieht.

```
HTTP/1.1 200 OK
x-amz-id-2: [id]
x-amz-request-id: [request_id]
```

```
Date: Wed, 01 Mar 2009 12:00:00 GMT
Content-Type: [type]
Content-Length: [length]
Connection: close
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<RequestPaymentConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Payer>Requester</Payer>
</RequestPaymentConfiguration>
```

Diese Antwort zeigt, dass der `payer`-Wert auf `Requester` gesetzt ist.

Herunterladen von Objekten aus Buckets mit Zahlung durch den Anforderer

Den Anforderern werden die Gebühren für das Herunterladen von Daten aus Buckets mit Zahlung durch den Anforderer in Rechnung gestellt. Deshalb müssen die Anforderungen einen speziellen Parameter enthalten, `x-amz-request-payer`, der bestätigt, dass der Anforderer weiß, dass er Gebühren für den Download zahlen muss. Um auf Objekte in Buckets mit Zahlung durch den Anforderer zuzugreifen, müssen die Anforderungen eine der folgenden Informationen enthalten.

- Für DELETE-, GET-, HEAD-, POST- und PUT-Anforderungen muss `x-amz-request-payer : requester` in den Header aufgenommen werden
- Für signierte URLs muss `x-amz-request-payer=requester` in die Anforderung aufgenommen werden

Ist die Anforderung erfolgreich und dem Auftraggeber werden Gebühren in Rechnung gestellt, enthält die Antwort den Header `x-amz-request-charged:requester`. Ist `x-amz-request-payer` nicht in der Anfrage enthalten, gibt Amazon S3 einen 403-Fehler zurück und stellt dem Bucket-Eigentümer die Gebühren für die Anfrage in Rechnung.

> **Note**
>
> Bucket-Eigentümer müssen ihren Anforderungen `x-amz-request-payer` nicht hinzufügen. Stellen Sie sicher, dass Sie `x-amz-request-payer` und seinen Wert in die Berechnung Ihrer Signatur aufgenommen haben. Weitere Informationen finden Sie unter Konstruieren des CanonicalizedAmzHeaders Elements .

Verwenden der REST-API

Herunterladen von Objekten aus einem Bucket mit Zahlung durch den Anforderer

- Verwenden Sie eine GET-Anforderung, um ein Objekt aus einem Bucket mit Zahlung durch den Anforderer herunterzuladen, wie in der folgenden Anforderung gezeigt.

```
GET / [destinationObject] HTTP/1.1
Host: [BucketName].s3.amazonaws.com
x-amz-request-payer : requester
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]
```

Ist die GET-Anforderung erfolgreich und dem Auftraggeber werden Gebühren in Rechnung gestellt, enthält die Antwort `x-amz-request-charged:requester`.

Amazon S3 kann für Anfragen, die versuchen, Objekte aus einem Bucket mit Zahlung durch den Anforderer abzurufen, einen `Access Denied`-Fehler zurückgeben. Weitere Informationen finden Sie unter Fehlermeldungen in der Amazon Simple Storage Service-API-Referenz.

Verwenden der AWS CLI

Um Objekte aus einem Bucket mit Zahlung durch den Anforderer über die herunterzuladen AWS CLI, geben Sie `--request-payer requester` als Teil Ihrer `-get-object`Anforderung an. Weitere Informationen finden Sie unter get-object in der AWS CLI -Referenz.

Beschränkungen und Einschränkungen von Buckets

Ein Amazon S3-Bucket gehört dem AWS-Konto , der ihn erstellt hat. Die Bucket-Eigentümerschaft ist nicht auf ein anderes Konto übertragbar.

Wenn Sie einen Bucket erstellen, wählen Sie seinen Namen und die aus, in der er erstellt AWS-Region werden soll. Name oder Region einmal erstellter Buckets können nicht nachträglich geändert werden.

Wenn Sie einen Bucket benennen, sollten Sie einen Namen wählen, der für Sie oder Ihr Unternehmen relevant ist. Vermeiden Sie die Verwendung von Namen, die mit anderen Entitäten verbunden sind. Zum Beispiel sollten Sie es vermeiden, `AWS` oder `Amazon` in Ihrem Bucket-Namen zu verwenden.

Standardmäßig können Sie bis zu 100 Buckets in jedem Ihrer erstellen AWS-Konten. Wenn Sie weitere Buckets benötigen, können Sie Ihr Konto-Bucket-Kontingent auf maximal 1.000 Buckets erhöhen, indem Sie eine Anfrage für eine Kontingenterhöhung senden. Es gibt keinen Leistungsunterschied, ungeachtet dessen, ob Sie viele Buckets oder nur wenige verwenden.

> **Note**
>
> Sie müssen nicht mehrere Anfragen zur Erhöhung des Kontingents für jedes einreichen AWS-Region. Ihr Bucket-Kontingent gilt für Ihr AWS-Konto.

Weitere Informationen über die Vorgehensweise zum Erhöhen des Bucket-Kontingents finden Sie unter AWS -Service-Quotas in der Allgemeinen AWS -Referenz.

Wiederverwenden von Bucket-Namen

Wenn ein Bucket leer ist, können Sie ihn löschen. Nachdem ein Bucket gelöscht wurde, wird der Name zur Wiederverwendung verfügbar. Nachdem Sie den Bucket gelöscht haben, können Sie den Namen aus verschiedenen Gründen möglicherweise jedoch nicht wiederverwenden.

Wenn Sie beispielsweise den Bucket löschen und der Name zur Wiederverwendung verfügbar wird, kann ein anderes AWS-Konto einen Bucket mit dem Namen erstellen. Außerdem kann einige Zeit vergehen, bis Sie den Namen eines gelöschten Buckets wiederverwenden können. Wenn Sie denselben Bucket-Namen verwenden möchten, empfehlen wir, den Bucket nicht zu löschen.

Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

Einschränkungen für Objekte und Buckets

Die maximale Bucket-Größe oder die Anzahl der Objekte, die Sie in einem Bucket speichern können, ist nicht begrenzt. Sie können alle Ihre Objekte in einem einzigen Bucket speichern, oder sie über mehrere Buckets verteilen. Sie können jedoch keinen Bucket über einen anderen Bucket erstellen.

Bucket-Operationen

Das Hochverfügbarkeits-Engineering von Amazon S3 konzentriert sich auf get-, put-, list- und delete-Operationen. Da Bucket-Operationen mit einem zentralisierten, globalen Ressourcenbereich arbeiten, ist es nicht empfehlenswert, Buckets auf dem Hochverfügbarkeits-Codepfad Ihrer Anwendung zu erstellen, zu löschen oder zu konfigurieren. Sinnvoller ist es, Buckets in einer separaten

Initialisierungs- oder Einrichtungsroutine zu erstellen, zu löschen oder zu konfigurieren, die seltener ausgeführt wird.

Bucket-Benennung und automatisch erstellte Buckets

Wenn Ihre Anwendung automatisch Buckets erstellt, wählen Sie ein Bucket-Namensschema, das wahrscheinlich keine Namenskonflikte verursacht. Stellen Sie sicher, dass Ihre Anwendungslogik einen anderen Bucket-Namen auswählt, wenn ein Bucket-Name bereits vergeben ist.

Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

Hochladen, Herunterladen und Arbeiten mit Objekten in Amazon S3

Um Ihre Daten in Amazon S3 zu speichern, arbeiten Sie mit Ressourcen, die als Buckets und Objekte bezeichnet werden. Ein Bucket ist ein Container für Objekte. Ein Objekt ist eine Datei und alle Metadaten, die diese Datei beschreiben.

Um ein Objekt in Amazon S3 zu speichern, erstellen Sie einen Bucket und laden das Objekt dann in einen Bucket hoch. Wenn sich das Objekt im Bucket befindet, können Sie es öffnen, herunterladen und kopieren. Wenn Sie ein Objekt oder einen Bucket nicht mehr benötigen, können Sie diese Ressourcen aufräumen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

> **Important**
>
> Wenn Sie in der Amazon-S3-Konsole Open (Öffnen) oder Dowload As (Herunterladen als) für ein Objekt auswählen, werden durch diese Operationen vorsignierte URLs erstellt. Für die Dauer von fünf Minuten ist Ihr Objekt für jeden zugänglich, der Zugriff auf diese vorsignierten URLs hat. Weitere Informationen über vorsignierte URLs finden Sie unter Verwenden vorsignierter URLs.

Mit Amazon S3 zahlen Sie nur für das, was Sie tatsächlich nutzen. Weitere Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon S3. Wenn Sie neuer Amazon-S3-Kunde sind, können Sie kostenlos mit Amazon S3 beginnen. Weitere Informationen finden Sie unter Kostenloses Kontingent für AWS.

Themen

- Arbeiten mit Objekt-Metadaten
- Objekte hochladen
- Hochladen und Kopieren von Objekten mit mehrteiligen Uploads
- Objekte kopieren
- Herunterladen von Objekten
- Überprüfung der Objektintegrität
- Löschen von Amazon-S3-Objekten
- Organisieren, Auflisten und Arbeiten mit Ihren Objekten
- Arbeiten mit vorsignierten URLs
- Transformieren von Objekten mit S3 Object Lambda

Übersicht über Amazon-S3-Objekte

Amazon S3 ist ein Objektspeicher, der eindeutige Schlüsselwerte verwendet, um so viele Objekte zu speichern, wie Sie möchten. Sie speichern diese Objekte in einem oder mehreren Buckets. Jedes Objekt kann eine Größe von bis zu 5 TB haben. Ein Objekt besteht aus Folgendem:

Schlüssel

Der Name, den Sie einem Objekt zuweisen. Zum Abrufen des Objekts verwenden Sie den Objektschlüssel. Weitere Informationen finden Sie unter Arbeiten mit Objektmetadaten.

Versions-ID

In einem Bucket wird ein Objekt durch einen Schlüssel und eine Versions-ID eindeutig identifiziert. Die Versions-ID ist eine Zeichenfolge, die Amazon S3 generiert, wenn Sie das Objekt einem Bucket hinzufügen. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Value

Der Inhalt, den Sie speichern.

Ein Objektwert kann eine beliebige Bytefolge sein. Objekte können von 0 bis 5 TB groß sein. Weitere Informationen finden Sie unter Objekte hochladen.

Metadaten

Ein Satz von Name-Wert-Paaren, mit dem Sie Informationen zum Objekt speichern können. Sie können Ihren Objekten in Amazon S3 Metadaten zuweisen, auch als benutzerdefinierte Metadaten bezeichnet. Amazon S3 weist diesen Objekten auch System-Metadatenhinzu, die es für die Verwaltung von Objekten verwendet. Weitere Informationen finden Sie unter Arbeiten mit Objektmetadaten.

Subressourcen

Amazon S3 verwendet den Subressourcen-Mechanismus, um objektspezifische zusätzliche Informationen zu speichern. Subressourcen sind Objekten untergeordnet, deshalb sind werden immer einer anderen Entity zugeordnet, wie beispielsweise einem Objekt oder einem Bucket. Weitere Informationen finden Sie unter Objekt-Subressourcen.

Informationen zur Zugriffskontrolle

Sie können den Zugriff auf Objekte kontrollieren, die Sie in Amazon S3 speichern. Amazon S3 unterstützt sowohl eine ressourcenbasierte Zugriffskontrolle, wie beispielsweise Access Control List (ACL), Bucket-Richtlinien und benutzerbasierte Zugriffskontrolle. Weitere Informationen über die Kontrolle des Zugriffs über finden Sie im:

- Bewährte Methoden für die Zugriffssteuerung
- Richtlinien für Zugriffsrichtlinien
- Identity and Access Management in Amazon S3
- Konfigurieren von ACLs

Ihre Amazon-S3-Ressourcen (wie z. B. Buckets und Objekte) sind standardmäßig privat. Sie müssen anderen ausdrücklich eine Berechtigung erteilen, damit sie auf diese Ressourcen zugreifen dürfen. Weitere Informationen über das Teilen von Objekten finden Sie unter Gemeinsame Nutzung von Objekten mit vorsignierten URLs.

Tags

Sie können Tags verwenden, um Ihre gespeicherten Objekte zu kategorisieren, für die Zugriffskontrolle oder die Kostenzuordnung. Weitere Informationen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

Objekt-Subressourcen

Amazon S3 definiert eine Menge von Subressourcen, die Buckets und Objekten zugeordnet sind. Subressourcen sind Objekten untergeordnet. Dies bedeutet, dass Subressourcen nicht alleine existieren. Sie sind immer einer anderen Entität zugeordnet, z. B. einem Objekt oder einem Bucket.

Die folgende Tabelle listet die Subressourcen auf, die Amazon-S3-Objekten zugeordnet sind.

Subressource	Beschreibung
acl	Enthält eine Liste der Rechte, wobei die Empfänger und die erteilten Berechtigungen identifizieren werden. Wenn Sie ein Objekt erstellen, identifiziert die `acl` den Objekteigentümer, der volle Kontrolle über das Objekt hat. Sie können eine Objekt-ACL abrufen oder sie durch eine aktualisierte Liste erteilter Rechte ersetzen. Bei einer Aktualisierung einer ACL müssen Sie die vorhandene ACL ersetzen. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht.

Erstellen von Objektschlüsselnamen

Der Objektschlüssel (oder Schlüsselname) identifiziert das Objekt in einem Amazon-S3-Bucket eindeutig. Objekt-Metadaten bestehen aus Name/Wert-Paaren. Weitere Informationen zu Objekt-Metadaten erhalten Sie unter Arbeiten mit Objekt-Metadaten.

Wenn Sie ein Objekt erstellen, geben Sie den Schlüsselnamen an, der das Objekt in dem Bucket eindeutig identifiziert. Wenn Sie in der Amazon-S3-Konsole einen Bucket markieren, wird beispielsweise eine Liste der Objekte in Ihrem Bucket angezeigt. Diese Namen sind die Objektschlüssel. Der Objektschlüsselname ist eine Sequenz aus Unicode-Zeichen mit UTF-8-Codierung und einer Länge von bis zu 1 024 Byte. Bei den Objektschlüsselnamen muss die Groß- und Kleinschreibung beachtet werden.

> **Note**
>
> Objektschlüsselnamen mit dem Wert „soap“ werden für virtual-hosted-style Anforderungen nicht unterstützt. Für Werte von Objektschlüsselnamen, bei denen „soap“ verwendet wird, muss stattdessen eine URL im Pfadformat verwendet werden.

Das Amazon-S3-Datenmodell ist eine flache Struktur: Sie erstellen einen Bucket und der Bucket speichert Objekte. Es gibt keine Hierarchie von Unterbuckets oder Unterordnern. Sie können jedoch mit den Schlüsselnamenpräfixen und Trennzeichen eine logische Hierarchie erschließen, wie in der Amazon-S3-Konsole. Die Amazon-S3-Konsole unterstützt ein Ordnerkonzept. Weitere Informationen zum Bearbeiten von Metadaten über die Amazon-S3-Konsole finden Sie unter Bearbeiten von Objektmetadaten in der Amazon-S3-Konsole.

Angenommen, Ihr Bucket (`admin-created`) enthält vier Objekte mit den folgenden Objektschlüsseln:

`Development/Projects.xls`

`Finance/statement1.pdf`

`Private/taxdocument.pdf`

`s3-dg.pdf`

Die Konsole verwendet die Schlüsselnamenpräfixe (`Development/`, `Finance/` und `Private/`) und das Trennzeichen ("/"), um eine Ordnerstruktur wie dargestellt anzuzeigen. Der `s3-dg.pdf`-Schlüssel hat kein Präfix, deshalb erscheint sein Objekt direkt auf Root-Ebene des Buckets. Wenn Sie den Ordner `Development/` öffnen, werden Sie darin das Objekt `Projects.xlsx` sehen.

- Amazon S3 unterstützt Buckets und Objekt. Es gibt keine Hierarchie in Amazon S3. Durch die Verwendung von Präfixen und Trennzeichen in einem Objektschlüsselnamen können die Amazon S3-Konsole und die AWS SDKs jedoch eine Hierarchie ableiten und das Ordnerkonzept einführen.
- Die Amazon-S3-Konsole implementiert die Ordnerobjekterstellung, indem sie ein Null-Byte-Objekt mit dem Wert des Ordners für Präfix und Trennzeichen als Schlüssel erstellt. Diese Ordnerobjekte werden nicht in der Konsole angezeigt. Andernfalls verhalten sie sich wie alle anderen Objekte und können über die REST-API, AWS CLI und AWS SDKs angezeigt und manipuliert werden.

Richtlinien für Objektschlüsselnamen

Sie können in einem Objektschlüsselnamen jedes beliebige UTF-8-Zeichen verwenden. Die Verwendung bestimmter Zeichen in Schlüsselnamen kann jedoch bei manchen Anwendungen und Protokollen zu Problemen führen. Die folgenden Richtlinien helfen Ihnen, die Compliance mit DNS, web-sicheren Zeichen, XML-Parsern und anderen APIs zu maximieren.

Sichere Zeichen

Die folgenden Zeichensätze sind allgemein sicher für die Verwendung in Schlüsselnamen.

Alphanumeric characters	• 0-9 • a-z • A-Z
Special characters	• Ausrufezeichen (`!`) • Bindestrich (`-`) • Unterstrich (`_`) • Punkt (`.`) • Sternchen (`*`) • Einzelnes Anführungszeichen (`'`) • Öffnende Klammer (`(`) • Schließende Klammer (`)`)

Nachfolgend finden Sie Beispiele für gültige Objektschlüsselnamen:

- `4my-organization`
- `my.great_photos-2014/jan/myvacation.jpg`
- `videos/2014/birthday/video1.wmv`

> **Note**
>
> Bei Objekten mit Schlüsselnamen, die mit Punkt(en) "." enden, die über die Amazon-S3-Konsole heruntergeladen werden, werden sämtliche Punkte "." aus dem Schlüsselnamen des heruntergeladenen Objekts entfernt. Um ein Objekt mit dem Schlüsselnamen herunterzuladen, der auf Punkt(e) „.“ endet, der im heruntergeladenen Objekt beibehalten wird, müssen Sie die AWS Command Line Interface (AWS CLI), AWS SDKs oder die REST-API verwenden.
> Darüber hinaus sollten Sie folgende Präfixbeschränkungen beachten:

- Objekte mit dem Präfix „./“ müssen mit der AWS Command Line Interface (AWS CLI), AWS SDKs oder der REST-API hoch- oder heruntergeladen werden. Sie können die Amazon-S3-Konsole nicht verwenden.
- Objekte mit dem Präfix „../“ können nicht mit AWS Command Line Interface (AWS CLI) oder der Amazon-S3-Konsole hochgeladen werden.

Zeichen, die möglicherweise eine Sonderverarbeitung benötigen

Die folgenden Zeichen in einem Schlüsselnamen erfordern möglicherweise eine zusätzliche Verarbeitung im Code oder müssen URL-codiert oder als HEX angegeben werden. Einige davon sind nicht darstellbare Zeichen, und Ihr Browser kann sie ggf. nicht verarbeiten, was zudem einer speziellen Vorgehensweise bedarf:

- Ampersand ("&")
- Dollar ("$")
- ASCII-Zeichenbereiche 00–1F hex (0–31 dezimal) und 7F (127 dezimal)
- 'At'-Symbol ("@")
- Gleichheitszeichen ("=")
- Semikolon (";")
- Schrägstrich ("/")
- Doppelpunkt (":")
- Plus ("+")
- Leerzeichen – Wichtige Leerzeichenfolgen gehen möglicherweise bei bestimmten Verwendungszwecken verloren (insbesondere Mehrfachleerzeichen).
- Komma (",")
- Fragezeichen ("?")

Zeichen, die Sie vermeiden sollten

Sie sollten in Schlüsselnamen die folgenden Zeichen vermeiden, weil sie einen maßgeblichen Arbeitsaufwand erfordern, um konsistent über alle Anwendungen zu sein.

- Umgekehrter Schrägstrich ("\")

- Linke geschweifte Klammer ("{")
- Nicht darstellbare ASCII-Zeichen (128-255 Dezimalzeichen)
- Caret ("^")
- Rechte geschweifte Klammer ("}")
- Prozentzeichen ("%")
- Accent Grave ("`")
- Rechte eckige Klammer ("]")
- Anführungszeichen
- Größersymbol (">")
- Linke eckige Klammer ("[")
- Tilde ("~")
- Kleiner als-Zeichen („<“)
- Pfundzeichen ("#")
- Vertikaler Strich ("|")

Schlüsselbeschränkungen für XML-bezogene Objekte

Wie im XML-Standard für die end-of-line Handhabung von angegeben, wird der gesamte XML-Text so normalisiert, dass einzelne Zeilenumschaltungen (ASCII-Code 13) und Zeilenumschaltungen unmittelbar gefolgt von einem Zeilenvorschub (ASCII-Code 10) durch ein einzelnes Zeilenvorschubzeichen ersetzt werden. Um das korrekte Parsen von Objektschlüsseln in XML-Anforderungen zu gewährleisten, müssen Zeilenumbrüche und andere Sonderzeichen durch den entsprechenden XML-Entitätscode ersetzt werden, wenn sie in XML-Markierungen eingefügt werden. Im Folgenden finden Sie eine Liste solcher Sonderzeichen und ihrer entsprechenden Entitätscodes:

- ' wie `'`
- ” wie `"`
- & wie `&`
- < wie `<`
- > wie `>`
- \r wie `
` oder `
`
- \n wie `
` oder `
`

Example

Das folgende Beispiel veranschaulicht die Verwendung eines XML-Entitätscodes als Ersatz für eine Zeilenumschaltung. Diese `DeleteObjects`-Anforderung löscht ein Objekt mit dem `key`-Parameter: `/some/prefix/objectwith\rcarriagereturn` (wobei\ r die Zeilenumschaltung ist).

```
<Delete xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Object>
    <Key>/some/prefix/objectwith&#13;carriagereturn</Key>
  </Object>
</Delete>
```

Arbeiten mit Objekt-Metadaten

Sie können Objekt-Metadaten in Amazon S3 zum Zeitpunkt des Hochladens des Objekts festlegen. Objekt-Metadaten bestehen aus Name/Wert-Paaren. Nachdem Sie das Objekt hochgeladen haben, können Sie Objekt-Metadaten nicht mehr ändern. Die einzige Methode, wie Sie Objekt-Metadaten ändern können, ist es, eine Kopie des Objekts anzulegen und die Metadaten festzulegen.

Wenn Sie ein Objekt erstellen, geben Sie auch den Schlüsselnamen an, der das Objekt in dem Bucket eindeutig identifiziert. Der Objektschlüssel (oder Schlüsselname) identifiziert das Objekt in einem Amazon-S3-Bucket eindeutig. Weitere Informationen finden Sie unter Erstellen von Objektschlüsselnamen.

Es gibt zweierlei Arten von Metadaten in Amazon S3: systemdefinierte und benutzerdefinierte Metadaten. In den folgenden Abschnitten finden Sie weitere Informationen zu systemdefinierten und benutzerdefinierten Metadaten. Weitere Informationen zum Bearbeiten von Metadaten mit der Amazon-S3-Konsole finden Sie unter Bearbeiten von Objektmetadaten in der Amazon-S3-Konsole.

Systemdefinierte Objektmetadaten

Amazon S3 verwaltet für jedes in einem Bucket gespeicherte Objekt einen Satz Systemmetadaten. Amazon S3 verarbeitet diese Systemmetadaten nach Bedarf. Beispielsweise verwaltet Amazon S3 das Erstellungsdatum eines Objekts und seine Größe als Metadaten und verwendet dies Information für die Objektverwaltung.

Es gibt zwei Kategorien von Systemmetadaten:

- Systemdefiniert – Metadaten wie beispielsweise das Erstellungsdatum eines Objekts werden vom System definiert. Dies bedeutet, dass nur Amazon S3 den Wert ändern kann.

- Benutzerdefiniert – Andere Systemmetadaten, wie beispielsweise die für das Objekt konfigurierte Speicherklasse und die Angabe, ob für das Objekt eine serverseitige Verschlüsselung aktiviert ist, sind Beispiele für Metadaten, deren Werte Sie definieren können. Wenn Ihr Bucket als Website konfiguriert ist, möchten Sie eine Seite möglicherweise irgendwann auf eine andere Seite oder zu einer externen URL umleiten. In diesem Fall ist eine Webseite ein Objekt in Ihrem Bucket. Amazon S3 speichert den Wert für die Seitenumleitung als Systemmetadaten, deren Wert Sie steuern können.

  Wenn Sie Objekte erstellen, können Sie die Werte dieser Systemmetadaten konfigurieren oder sie nach Bedarf aktualisieren. Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.

  Amazon S3 verwendet - AWS KMS Schlüssel, um Ihre Amazon S3-Objekte zu verschlüsseln. AWS KMS verschlüsselt nur die Objektdaten. Die Prüfsumme wird zusammen mit dem angegebenen Algorithmus als Teil der Metadaten des Objekts gespeichert. Wenn für das Objekt eine serverseitige Verschlüsselung angefordert wird, wird die Prüfsumme in verschlüsselter Form gespeichert. Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter Datenschutz durch Verschlüsselung.

> **Note**
>
> Der PUT-Anforderungs-Header ist auf eine Größe von 8 KB begrenzt. Innerhalb des PUT-Anforderungs-Headers sind die systemdefinierten Metadaten auf eine Größe von 2 KB begrenzt. Die Größe der systemdefinierten Metadaten wird anhand der Summe der Byteanzahl der US-ASCII-Codierung jedes Schlüssels und Werts gemessen.

Die folgenden Tabelle enthält eine Liste der vom System definierten Metadaten, und gibt an, ob Sie sie aktualisieren können.

Name	Beschreibung	Kann der Benutzer den Wert ändern?
`Date`	Aktuelles Datum und Uhrzeit	Nein

Name	Beschreibung	Kann der Benutzer den Wert ändern?
`Cache-Control`	Ein allgemeines Header-Feld zum Angeben von Caching-Richtlinien.	Ja
`Content-Disposition`	Darstellungsinformationen zum Objekt.	Ja
`Content-Length`	Die Objektgröße in Bytes	Nein
`Content-Type`	Der Typ des Objekts.	Ja
`Last-Modified`	Das Datum, an dem das Objekt erstellt wurde, oder das letzte Änderungsdatum, je nachdem, welcher Wert aktueller ist Bei mehrteiligen Uploads entspricht das Objekterstellungsdatum dem Startdatum des mehrteiligen Uploads.	Nein
`ETag`	Das Entity-Tag (ETag). das eine bestimmte Version eines Objekts darstellt. Für Objekte, die nicht als mehrteiliger Upload hochgeladen werden und entweder unverschlüsselt oder durch serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt sind, ist der ETag ein MD5-Digest der Daten.	Nein
`x-amz-server-side-encryption`	Ein Header, der angibt, ob die serverseitige Verschlüsselung für das Objekt aktiviert ist und ob diese Verschlüsselung die AWS Key Management Service (AWS KMS)-Schlüssel (SSE-KMS) oder die von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) verwendet. Weitere Informationen finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung.	Ja

Name	Beschreibung	Kann der Benutzer den Wert ändern?
`x-amz-checksum-crc32` , `x-amz-checksum-crc32c` , `x-amz-checksum-sha1` , `x-amz-checksum-sha256`	Header, die die Prüfsumme oder den Digest des Objekts enthalten. Je nach Prüfsummenalgorithmus, den Amazon S3 verwenden soll, wird höchstens jeweils einer dieser Header festgelegt. Weitere Informationen zur Auswahl des Prüfsummenalgorithmus finden Sie unter Überprüfung der Objektintegrität.	Nein
`x-amz-version-id`	Die Objekt-Versionsverwaltung. Wenn Sie die Versionsverwaltung für einen Bucket aktivieren, weist Amazon S3 allen Objekten, die dem Bucket hinzugefügt werden, eine Versions-ID zu. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.	Nein
`x-amz-delete-marker`	Ein boolesches Kennzeichen, das angibt, ob das Objekt eine Löschmarkierung ist. Dieses Kennzeichen wird nur in Buckets verwendet, für die die Versionsverwaltung aktiviert ist.	Nein
`x-amz-storage-class`	Die Speicherklasse, für die Speicherung des Objekts verwendet wird. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.	Ja
`x-amz-website-redirect-location`	Ein Header, der Anfragen für das jeweilige Objekt auf ein anderes Objekt im selben Bucket oder zu einer externen URL umleitet. Weitere Informationen finden Sie unter (Optional) Konfigurieren einer Webseitenumleitung.	Ja

Name	Beschreibung	Kann der Benutzer den Wert ändern?
`x-amz-server-side-encryption-aws-kms-key-id`	Ein Header, der die ID des KMS-Schlüssels mit AWS KMS symmetrischer Verschlüsselung angibt, der zum Verschlüsseln des Objekts verwendet wurde. Dieser Header wird nur verwendet, wenn der Header `x-amz-server-side-encryption` vorhanden ist und den Wert `aws:kms` aufweist.	Ja
`x-amz-server-side-encryption-customer-algorithm`	Ein Header, der angibt, ob die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) aktiviert ist. Weitere Informationen finden Sie unter Verwenden von serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C).	Ja
`x-amz-tagging`	Das Tag-Set für das Objekt. Das Tag-Set muss als URL-Abfrageparameter codiert sein.	Ja

Benutzerdefinierte Objektmetadaten

Wenn Sie ein Objekt hochladen, können Sie ihm Metadaten zuweisen. Sie geben diese optionalen Informationen als Name-Wert-Paar (Schlüssel-Wert) an, wenn Sie eine `PUT`- oder `POST`-Anforderung senden, um das Objekt zu erstellen. Wenn Sie Objekte mit der REST-API hochladen, müssen die optionalen benutzerdefinierten Metadatennamen mit `x-amz-meta-` beginnen, damit sie von den anderen HTTP-Headern unterschieden werden können. Wenn Sie das Objekt mit der REST-API abrufen, wird dieses Präfix zurückgegeben. Wenn Sie Objekte mit der SOAP API hochladen, ist das Präfix nicht erforderlich. Wenn Sie das Objekt mit der SOAP API abrufen, wird das Präfix entfernt, unabhängig davon, welche API Sie zum Hochladen des Objekts verwendet haben.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Anstatt

SOAP zu verwenden, empfehlen wir, entweder die REST-API oder die AWS SDKs zu verwenden.

Wenn Metadaten über die REST-API abgerufen werden, kombiniert Amazon S3 Header, die denselben Namen haben (wobei die Groß-/Kleinschreibung ignoriert wird) in einer durch Kommata getrennten Liste. Wenn einige Metadaten nicht darstellbare Zeichen enthalten, werden sie nicht zurückgegeben. Stattdessen wird der `x-amz-missing-meta`-Header mit der Anzahl der nicht darstellbaren Metadateneinträge als Wert zurückgegeben. Die `HeadObject`-Aktion ruft Metadaten von einem Objekt ab, ohne das Objekt selbst zurückzugeben. Dieser Vorgang ist nützlich, wenn Sie nur an den Metadaten eines Objekts interessiert sind. Um `HEAD` verwenden zu können, müssen Sie `READ`-Zugriff auf das Objekt haben. Weitere Informationen finden Sie unter HeadObject in der API-Referenz zu Amazon Simple Storage Service.

Benutzerdefinierte Metadaten sind Schlüssel/Wert-Paare. Amazon S3 speichert benutzerdefinierte Metadatenschlüssel in Kleinbuchstaben.

Amazon S3 erlaubt beliebige Unicode-Zeichen in Ihren Metadatenwerten.

Um Probleme bei der Darstellung dieser Metadatenwerte zu vermeiden, sollten Sie US-ASCII-Zeichen verwenden, wenn REST verwendet wird, und UTF-8-Zeichen, wenn SOAP oder browserbasierte Uploads über `POST` verwendet werden.

Wenn Sie Nicht-US-ASCII-Zeichen in Ihren Metadatenwerten verwenden, wird die bereitgestellte Unicode-Zeichenfolge auf Nicht-US-ASCII-Zeichen untersucht. Werte solcher Header werden gemäß RFC 2047 vor dem Speichern zeichendecodiert und gemäß RFC 2047 codiert, um sie für den Datentransfer sicher zu machen, bevor sie zurückgegeben werden. Wenn die Zeichenfolge nur US-ASCII Zeichen enthält, wird sie so dargestellt, wie sie ist.

Im Folgenden wird ein Beispiel gezeigt.

```
PUT /Key HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-nonascii: ÄMÄZÕÑ S3

HEAD /Key HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-nonascii: =?UTF-8?B?w4PChE3Dg8KEWsODwpXDg8KRIFMz?=

PUT /Key HTTP/1.1
```

```
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3

HEAD /Key HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3
```

> **Note**
>
> Der PUT-Anforderungs-Header ist auf eine Größe von 8 KB begrenzt. Innerhalb des PUT-Anforderungs-Headers sind die benutzerdefinierten Metadaten auf eine Größe von 2 KB begrenzt. Die Größe der benutzerdefinierten Metadaten wird anhand der Summe der Byteanzahl der UTF-8-Codierung jedes Schlüssels und Werts gemessen.

Informationen zum Ändern der Metadaten Ihres Objekts, nachdem das Objekt hochgeladen wurde, indem eine Kopie des Objekts erstellt und geändert und das alte Objekt ersetzt wurde oder indem eine neue Version erstellt wurde, finden Sie unter Bearbeiten von Objektmetadaten in der Amazon-S3-Konsole.

Bearbeiten von Objektmetadaten in der Amazon-S3-Konsole

Sie können die Amazon-S3-Konsole verwenden, um Metadaten von vorhandenen S3-Objekten zu bearbeiten. Einige Metadaten werden von Amazon S3 festgelegt, wenn Sie das Objekt hochladen. Beispielsweise sind `Content-Length` und `Last-Modified` systemdefinierte Objektmetadatenfelder, die von einem Benutzer nicht geändert werden können.

Sie können einige Metadaten festlegen, wenn Sie das Objekt hochladen. Es kann danach je nach Anforderungen bearbeitet werden. Beispielsweise haben Sie vielleicht eine Reihe von Objekten, die Sie ursprünglich in der `STANDARD`-Speicherklasse gespeichert haben. Im Laufe der Zeit benötigen Sie diese Daten möglicherweise nicht mehr, um hochverfügbar zu sein. Sie können also die Speicherklasse in `GLACIER` ändern, indem Sie den Wert des `x-amz-storage-class`-Schlüssels von `STANDARD` nach `GLACIER` bearbeiten.

> **Note**
>
> Berücksichtigen Sie die folgenden Probleme, wenn Sie Objektmetadaten in Amazon S3 bearbeiten:

- Dieser Vorgang erstellt eine Kopie des Objekts mit aktualisierten Einstellungen und dem Datum der letzten Änderung. Wenn S3-Versioning aktiviert ist, wird eine neue Version des Objekts erstellt, und das vorhandene Objekt wird zu einer älteren Version. Wenn das S3-Versioning nicht aktiviert ist, ersetzt eine neue Kopie des Objekts das ursprüngliche Objekt. Das , das der IAM-Rolle AWS-Konto zugeordnet ist, die die Eigenschaft ändert, wird auch Besitzer des neuen Objekts oder (Objektversion).
- Das Bearbeiten von Metadaten aktualisiert die Werte für vorhandene Schlüsselnamen.
- Objekte, die mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) verschlüsselt wurden, können nicht mit der Konsole kopiert werden. Sie müssen die AWS CLI, AWS das SDK oder die Amazon S3-REST-API verwenden.

Warning

Warten Sie beim Bearbeiten von Metadaten von Ordnern, bis der Vorgang `Edit metadata` abgeschlossen ist, bevor Sie dem Ordner neue Objekte hinzufügen. Andernfalls könnten auch neue Objekte bearbeitet werden.

In den folgenden Themen wird beschrieben, wie Metadaten eines Objekts mithilfe der Amazon-S3-Konsole bearbeitet werden.

Bearbeiten von systemdefinierten Metadaten

Sie können verschiedene jedoch nicht alle System-Metadaten für ein S3-Objekt konfigurieren. Eine Liste der systemdefinierten Metadaten und ob Sie ihre Werte ändern können, finden Sie unter Systemdefinierte Objektmetadaten.

Bearbeiten systemdefinierter Metadaten eines Objekts

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Navigieren Sie zu Ihrem Amazon-S3-Bucket oder -Ordner und aktivieren Sie das Kontrollkästchen links neben den Namen der Objekte mit Metadaten, die Sie bearbeiten möchten.
3. Klicken Sie im Menü Actions (Aktionen) auf Edit actions (Aktionen bearbeiten) und wählen Sie Edit metadata (Metadaten bearbeiten).

4. Überprüfen Sie die aufgeführten Objekte und wählen Sie Add metadata (Metadaten hinzufügen).
5. Wählen Sie für Type(Typ) der Metadaten, System-defined (systemdefiniert) aus.
6. Geben Sie einen eindeutigen Key (Schlüssel) und den Value (Wert) der Metadaten an.
7. Um weitere Metadaten zu bearbeiten, wählen Sie Add metadata (Metadaten hinzufügen). Sie können auch Entfernen wählen, um einen Satz von zu entfernen type-key-values.
8. Wenn Sie fertig sind, wählen Sie Edit metadata (Metadaten ändern). Amazon S3 ändert daraufhin die Metadaten der angegebenen Objekte.

Bearbeiten von benutzerdefinierten Metadaten

Sie können benutzerdefinierte Metadaten eines Objekts bearbeiten, indem Sie das Metadatenpräfix `x-amz-meta-` und einen Namen miteinander kombinieren, um einen benutzerdefinierten Schlüssel zu generieren. Wenn Sie beispielsweise den benutzerdefinierten Namen `alt-name` hinzufügen, wäre der Metadaten-Schlüssel `x-amz-meta-alt-name`.

Benutzerdefinierte Metadaten können insgesamt bis zu 2 KB umfassen. Um die Gesamtgröße von benutzerdefinierten Metadaten zu berechnen, addieren Sie die Anzahl der Bytes in der UTF-8-Codierung für jeden Schlüssel und Wert. Sowohl die Schlüssel als auch deren Werte müssen US-ASCII-Standards entsprechen. Weitere Informationen finden Sie unter Benutzerdefinierte Objektmetadaten.

So bearbeiten Sie benutzerdefinierte Metadaten eines Objekts

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der die Objekte enthält, denen Sie Metadaten hinzufügen möchten.

   Sie können auch optional zu einem Ordner navigieren.
3. Aktivieren Sie in der Liste Objects (Objekte) das Kontrollkästchen neben den Namen der Objekte, denen Sie Metadaten hinzufügen möchten.
4. Wählen Sie im Menü Actions (Aktionen) die Option Edit metadata (Metadaten bearbeiten).
5. Überprüfen Sie die aufgeführten Objekte und wählen Sie Add metadata (Metadaten hinzufügen).
6. Wählen Sie für Type (Typ) der Metadaten, User-defined (benutzerdefiniert) aus.
7. Geben Sie einen auf `x-amz-meta-` folgenden eindeutigen, benutzerdefinierten Key (Schlüssel) ein. Geben Sie ebenfalls einen Value (Wert) der Metadaten ein.

8. Um zusätzliche Metadaten hinzuzufügen, wählen Sie Add metadata (Metadaten hinzufügen). Sie können auch Entfernen wählen, um einen Satz von zu entfernen type-key-values.
9. Wählen Sie Edit metadata (Metadaten bearbeiten).

   Amazon S3 bearbeitet die Metadaten der angegebenen Objekte.

Objekte hochladen

Wenn Sie eine Datei Amazon S3 hochladen, wird sie als S3-Objekt gespeichert. Objekte umfassen die Datei und die Metadaten, die das Objekt beschreiben. Ein Bucket kann eine unbegrenzte Anzahl von Objekten aufnehmen. Bevor Sie Dateien zu Amazon S3 hochladen können, benötigen Sie die Schreibberechtigungen für den Bucket. Weitere Informationen zu den Zugriffsberechtigungen finden Sie unter Identity and Access Management in Amazon S3.

Sie können beliebige Dateitypen – Bilder, Backups, Daten, Filme usw. – in einen S3-Bucket hochladen. Die maximale Größe einer Datei, die Sie über die Amazon-S3-Konsole hochladen können, ist 160 GB. Um eine Datei mit mehr als 160 GB hochzuladen, verwenden Sie die AWS Command Line Interface (AWS CLI), AWS SDKs oder die Amazon S3-REST-API.

Wenn Sie ein Objekt mit einem Schlüsselnamen hochladen, der bereits in einem Bucket mit aktiviertem Versioning vorhanden ist, erstellt Amazon S3 eine weitere Version des Objekts, statt das vorhandene Objekt zu ersetzen. Weitere Informationen über Versioning finden Sie unter Verwenden der S3-Konsole.

Abhängig von der Größe der Daten, die Sie hochladen, bietet Amazon S3 die folgenden Optionen:

- Hochladen eines Objekts in einer einzigen Operation mithilfe der - AWS SDKs , der REST-API oder AWS CLI – Mit einer einzigen PUT Operation können Sie ein einzelnes Objekt mit einer Größe von bis zu 5 GB hochladen.
- Hochladen eines einzelnen Objekts mit der Amazon-S3-Konsole – Mit der Amazon-S3-Konsole können Sie ein einzelnes Objekt mit einer Größe von bis zu 160 GB hochladen.
- Hochladen eines Objekts in Teilen mithilfe der - AWS SDKs , der REST-API oder AWS CLI – Mit der API-Operation für mehrteilige Uploads können Sie ein einzelnes großes Objekt mit einer Größe von bis zu 5 TB hochladen.

  Die API-Operation für mehrteilige Uploads ist darauf ausgelegt, die Upload-Leistung für größere Objekte zu verbessern. Sie können ein Objekt in Teilen hochladen. Diese Objektteile können unabhängig, in jeder beliebigen Reihenfolge und parallel hochgeladen werden. Sie können einen

mehrteiligen Upload verwenden, um Objekte mit einer Größe von 5 MB bis 5 TB hochzuladen. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Wenn Sie ein Objekt hochladen, wird das Objekt standardmäßig automatisch mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt. Wenn Sie es herunterladen, wird das Objekt entschlüsselt. Weitere Informationen finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets und Datenschutz durch Verschlüsselung.

Wenn Sie beim Hochladen eines Objekts einen anderen Standardverschlüsselungstyp verwenden möchten, können Sie in Ihren S3-PUT-Anforderungen auch die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) angeben oder die Standardverschlüsselungskonfiguration im Ziel-Bucket so festlegen, dass SSE-KMS zum Verschlüsseln Ihrer Daten verwendet wird. Weitere Informationen zu SSE-KMS finden Sie unter Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS). Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen. Weitere Informationen zu kontoübergreifenden Berechtigungen für KMS-Schlüssel finden Sie unter Erstellen von KMS-Schlüsseln, die von anderen Konten verwendet werden können im Entwicklerhandbuch zu AWS Key Management Service .

Wenn in Amazon S3 ein Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) auftritt, finden Sie weitere Informationen zu den häufigsten Ursachen unter Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3 .

Verwenden der S3-Konsole

In diesem Verfahren wird erläutert, wie Sie Objekte und Ordner mithilfe der Konsole in einen Amazon-S3-Bucket hochladen.

Wenn Sie ein Objekt hochladen, ist der Name des Objektschlüssels der Dateiname und etwaige optionale Präfixe. In der Amazon-S3-Konsole können Sie Ordner erstellen, um Ihre Objekte zu organisieren. In Amazon S3 werden Ordner als Präfixe dargestellt, die im Namen des Objektschlüssels angezeigt werden. Wenn Sie ein einzelnes Objekt in einen Ordner in der Amazon-S3-Konsole hochladen, wird der Ordnername im Namen des Objektschlüssels aufgenommen.

Wenn Sie beispielsweise ein Objekt mit dem Namen `sample1.jpg` in einen Ordner namens `backup` hochladen, lautet der Schlüsselname `backup/sample1.jpg`. Allerdings wird das Objekt in der

Konsole als `sample1.jpg` im Ordner `backup` angezeigt. Weitere Informationen zu Schlüsselnamen finden Sie unter Arbeiten mit Objekt-Metadaten.

> **Note**
>
> Wenn Sie ein Objekt umbenennen oder eine der Eigenschaften in der Amazon-S3-Konsole ändern, z. B. Speicherklasse, Verschlüsselung oder Metadaten, wird ein neues Objekt erstellt, das das alte ersetzt. Wenn S3-Versioning aktiviert ist, wird eine neue Version des Objekts erstellt, und das vorhandene Objekt wird zu einer älteren Version. Die Rolle, die die Eigenschaft ändert, wird auch Besitzer des neuen Objekts (oder der neuen Objektversion).

Wenn Sie einen Ordner hochladen, lädt Amazon S3 alle Dateien und Unterordner aus dem angegebenen Ordner in Ihren Bucket hoch. Es wird dann ein Objektschlüsselname zugewiesen, der eine Kombination aus dem hochgeladenen Dateinamen und dem Ordnernamen ist. Wenn Sie beispielweise einen Ordner mit dem Namen `/images` hochladen, der zwei Dateien, `sample1.jpg` und `sample2.jpg`, enthält, lädt Amazon S3 die Dateien hoch und weist ihnen die jeweiligen Schlüsselnamen `images/sample1.jpg` und `images/sample2.jpg` zu. Der Schlüsselname enthält den Ordnernamen als Präfix. Die Amazon-S3-Konsole zeigt nur den Teil des Schlüsselnamens nach dem letzten `/` an. Beispielsweise werden in einem `images`-Ordner die Objekte `images/sample1.jpg` und `images/sample2.jpg` als `sample1.jpg` und `sample2.jpg` angezeigt.

So laden Sie Ordner und Dateien in einen S3-Bucket hoch

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, in den Ihre Ordner oder Dateien hochgeladen werden sollen.
4. Klicken Sie auf Upload.
5. Führen Sie im Fenster Upload einen der folgenden Schritte aus:
   - Ziehen Sie Dateien und Ordner in das Fenster Upload (Hochladen) und legen Sie sie dort ab.
   - Wählen Sie Datei hinzufügen oder Ordner hinzufügen aus, wählen Sie die hochzuladenden Dateien oder Ordner und anschließend Öffnen aus.

6. Um die Versioning zu aktivieren, wählen Sie unter Destination (Ziel) die Option Enable Bucket Versioning (Bucket-Versioning aktivieren) aus.
7. Um die aufgelisteten Dateien und Ordner hochzuladen, ohne zusätzliche Upload-Optionen zu konfigurieren, wählen Sie unten auf der Seite Upload aus.

   Amazon S3 lädt Ihre Objekte und Ordner hoch. Wenn der Upload abgeschlossen ist, wird auf der Seite Upload: Status eine Erfolgsmeldung angezeigt.

So konfigurieren Sie zusätzliche Objekteigenschaften

1. Um Zugriffssteuerungslisten-Berechtigungen zu ändern, wählen Sie Permissions (Berechtigungen) aus.
2. Bearbeiten Sie die Berechtigungen unter Access control list (ACL) (Zugriffssteuerungsliste).

   Weitere Informationen zu den Zugriffsberechtigungen für Objekte finden Sie unter Festlegen von ACL-Berechtigungen für ein Objekt mit der S3-Konsole. Sie können der Öffentlichkeit Lesezugriff auf Ihre Objekte erteilen, und zwar für alle Dateien, die Sie hochladen. Wir empfehlen allerdings, die Standardeinstellung für den öffentlichen Lesezugriff nicht zu ändern. Die Erteilung von öffentlichem Lesezugriff ist nur für wenige Anwendungsfälle sinnvoll, beispielsweise, wenn Buckets für Websites verwendet werden. Nachdem Sie das Objekt hochgeladen haben, können Sie jederzeit Änderungen an den Objektberechtigungen vornehmen.
3. Um andere zusätzliche Eigenschaften zu konfigurieren, wählen Sie Properties (Eigenschaften) aus.
4. Wählen Sie im Abschnitt Speicherklasse die Speicherklasse für die Dateien aus, die Sie hochladen.

   Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.
5. Um die Verschlüsselungseinstellungen für Ihre Objekte zu aktualisieren, gehen Sie unter Server-side encryption settings (Serverseitige Verschlüsselungseinstellungen) wie folgt vor.

   a. Klicken Sie auf Specify an encryption key (Verschlüsselungsschlüssel angeben).

   b. Wählen Sie unter Verschlüsselungseinstellungen die Option Verwenden von Bucket-Einstellungen für die Standardverschlüsselung oder Überschreiben der Bucket-Einstellungen für die Standardverschlüsselung aus.

c. Wenn Sie Überschreiben der Bucket-Einstellungen für die Standardverschlüsselung ausgewählt haben, müssen Sie die folgenden Verschlüsselungseinstellungen konfigurieren.

- Um die hochgeladenen Dateien mit Schlüsseln zu verschlüsseln, die von Amazon S3 verwaltet werden, wählen Sie Von Amazon S3 verwalteter Schlüssel (SSE-S3) aus.

  Weitere Informationen finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

- Um die hochgeladenen Dateien mit Schlüsseln zu verschlüsseln, die in AWS Key Management Service (AWS KMS) gespeichert sind, wählen Sie AWS Key Management Service Schlüssel (SSE-KMS). Wählen Sie dann eine der folgenden Optionen für AWS KMS -Schlüssel aus:

  - Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS keys wählen und dann den KMS-Schlüssel in der Liste der verfügbaren Schlüssel aus.

    Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS -Schlüssel im Entwicklerhandbuch zu AWS Key Management Service .

  - Um den KMS-Schlüssel-ARN einzugeben, wählen Sie AWS KMS key ARN eingeben und geben Sie dann Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.

  - Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

    Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

    Important

    Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket verfügbar sind. Die Amazon-S3-Konsole führt nur die ersten 100 KMS-Schlüssel auf, die in derselben Region wie der Bucket verfügbar sind. Wenn Sie einen KMS-Schlüssel verwenden möchten, der nicht aufgeführt ist, müssen Sie den KMS-Schlüssel-ARN eingeben. Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über

die Berechtigung zum Verwenden des Schlüssels verfügen und Sie müssen den KMS-Schlüssel-ARN eingeben.
Amazon S3 unterstützt nur symmetrisch verschlüsselte KMS-Schlüssel und keine asymmetrischen KMS-Schlüssel. Weitere Informationen finden Sie unter Erkennen von symmetrischen und asymmetrischen KMS-Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service .

6. Wählen Sie On (Ein) aus, um zusätzliche Prüfsummen zu verwenden. Wählen Sie dann für Checksum function (Prüfsummenfunktion) die Funktion aus, die Sie verwenden möchten. Amazon S3 berechnet und speichert den Prüfsummenwert, nachdem es das gesamte Objekt erhalten hat. Sie können das Feld Precalculated value (Vorberechneter Wert) verwenden, um einen vorberechneten Wert anzugeben. In diesem Fall vergleicht Amazon S3 den von Ihnen angegebenen Wert mit dem eigenen berechneten Wert. Wenn die beiden Werte nicht übereinstimmen, generiert Amazon S3 einen Fehler.

   Mit zusätzlichen Prüfsummen können Sie den Prüfsummenalgorithmus angeben, den Sie zur Überprüfung Ihrer Daten verwenden möchten. Weitere Informationen zu zusätzlichen Prüfsummen finden Sie unter Überprüfung der Objektintegrität.

7. Um allen hochgeladenen Objekten Markierungen hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Geben Sie einen Tag-Namen in das Feld Schlüssel ein. Geben Sie einen Wert für das Tag ein.

   Das Markieren von Objekten ermöglicht Ihnen, Speicher zu kategorisieren. Jeder Tag ist ein Schlüssel/Wert-Paar. Bei Schlüssel- und Tag-Werten wird die Groß-/Kleinschreibung berücksichtigt. Sie können über bis zu 10 Markierungen pro Objekt verfügen. Ein Tag-Schlüssel kann maximal 128 Unicode-Zeichen und die Tag-Werte bis zu 255 Unicode-Zeichen lang sein. Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

8. Um Metadaten hinzuzufügen, wählen Sie Add metadata (Metadaten hinzufügen).

   a. Wählen Sie unter Typdie Option System defined (System definiert) oder User defined (Benutzerdefiniert) aus.

      Für systemdefinierte Metadaten können Sie gemeinsame HTTP-Header wie Content-Type und Content-Disposition auswählen. Eine Liste systemdefinierter Metadaten und Informationen dazu, ob Sie Werte hinzufügen können, finden Sie unter Systemdefinierte Objektmetadaten. Metadaten, die mit dem Präfix `x-amz-meta-` beginnen, werden als

benutzerdefinierte Metadaten behandelt. Benutzerdefinierte-Metadaten werden mit dem Objekt gespeichert und beim Download mit ihm zurückgegeben. Sowohl die Schlüssel als auch deren Werte müssen US-ASCII-Standards entsprechen. Benutzerdefinierte Metadaten können bis zu 2 KB umfassen. Weitere Informationen zu systemdefinierten und benutzerdefinierten Metadaten finden Sie unter Arbeiten mit Objekt-Metadaten.

b. Wählen Sie für Key einen Schlüssel aus.

c. Geben Sie einen Wert für den Schlüssel ein.

9. Um Ihre Objekte hochzuladen, wählen Sie Upload (Hochladen) aus.

   Amazon S3 lädt Ihr Objekt hoch. Wenn der Upload abgeschlossen ist, wird auf der Seite Upload: status eine Erfolgsmeldung angezeigt.

10. Wählen Sie Exit (Beenden) aus.

Verwenden der AWS SDKs

Sie können die - AWS SDKs verwenden, um Objekte in Amazon S3 hochzuladen. Die SDKs stellen Wrapper-Bibliotheken für den einfachen Upload von Daten bereit. Weitere Informationen finden Sie in der Liste der unterstützten SDKs.

Dies sind Beispiele mit einigen ausgewählten SDKs:

.NET

Das folgende C#-Codebeispiel erstellt zwei Objekte mit zwei `PutObjectRequest`-Anforderungen:

- Die erste `PutObjectRequest`-Anforderung speichert als Beispielobjektdaten eine Textzeichenfolge. Sie gibt zudem Bucket- und Objektschlüsselnamen an.
- Die zweite `PutObjectRequest`-Anforderung lädt eine Datei durch Angabe des Dateinamens hoch. Diese Anforderung gibt außerdem den `ContentType`-Header und optionale Objektmetadaten (einen Titel) ein.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
```

```
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadObjectTest
    {
        private const string bucketName = "*** bucket name ***";
        // For simplicity the example creates two objects from the same file.
        // You specify key names for these objects.
        private const string keyName1 = "*** key name for first object created ***";
        private const string keyName2 = "*** key name for second object created
 ***";
        private const string filePath = @"*** file path ***";
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.EUWest1;

        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            WritingAnObjectAsync().Wait();
        }

        static async Task WritingAnObjectAsync()
        {
            try
            {
                // 1. Put object-specify only key name for the new object.
                var putRequest1 = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName1,
                    ContentBody = "sample text"
                };

                PutObjectResponse response1 = await
 client.PutObjectAsync(putRequest1);

                // 2. Put the object-set ContentType and add metadata.
                var putRequest2 = new PutObjectRequest
                {
                    BucketName = bucketName,
```

```
                    Key = keyName2,
                    FilePath = filePath,
                    ContentType = "text/plain"
                };

                putRequest2.Metadata.Add("x-amz-meta-title", "someTitle");
                PutObjectResponse response2 = await
 client.PutObjectAsync(putRequest2);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an
 object"
                        , e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Unknown encountered on server. Message:'{0}' when writing an
 object"
                    , e.Message);
            }
        }
    }
}
```

Java

Das folgende Beispiel erstellt zwei Objekte. Das erste Objekt besitzt eine Textzeichenfolge als Daten, und das zweite Objekt ist eine Datei. Das Beispiel erstellt das erste Objekt durch Angabe des Bucket-Namen, des Objektschlüssels und von Textdaten direkt in einem Aufruf an `AmazonS3Client.putObject()`. Das Beispiel erstellt das zweite Objekt mittels einer `PutObjectRequest`, die den Bucket-Namen, den Objektschlüssel und den Dateipfad angibt. Die `PutObjectRequest` gibt zudem den `ContentType`-Header und Titelmetadaten an.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.regions.Regions;
```

```
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectMetadata;
import com.amazonaws.services.s3.model.PutObjectRequest;

import java.io.File;
import java.io.IOException;

public class UploadObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String stringObjKeyName = "*** String object key name ***";
        String fileObjKeyName = "*** File object key name ***";
        String fileName = "*** Path to file to upload ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .build();

            // Upload a text string as a new object.
            s3Client.putObject(bucketName, stringObjKeyName, "Uploaded String
 Object");

            // Upload a file as a new object with ContentType and title specified.
            PutObjectRequest request = new PutObjectRequest(bucketName,
 fileObjKeyName, new File(fileName));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentType("plain/text");
            metadata.addUserMetadata("title", "someTitle");
            request.setMetadata(metadata);
            s3Client.putObject(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
```

```
            e.printStackTrace();
        }
    }
}
```

JavaScript

Im folgenden Beispiel wird eine vorhandene Datei in einen Amazon-S3-Bucket in einer spezifischen Region hochgeladen.

```
import { PutObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new PutObjectCommand({
    Bucket: "test-bucket",
    Key: "hello-s3.txt",
    Body: "Hello S3!",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

PHP

Dieses Beispiel führt Sie durch die Verwendung von Klassen aus dem AWS SDK for PHP , um ein Objekt mit einer Größe von bis zu 5 GB hochzuladen. Für größere Dateien müssen Sie eine API-Operation für einen mehrteiligen Upload verwenden. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Dieses Beispiel setzt voraus, dass Sie die Anweisungen für Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen befolgen und das AWS SDK for PHP ordnungsgemäß installiert ist.

Example – Erstellen eines Objekts in einem Amazon-S3-Bucket, indem Daten hochgeladen werden

Das folgende PHP- Beispiel erstellt ein Objekt in einem spezifizierten Bucket, indem die Daten mit der `putObject()`-Methode hochgeladen werden. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen.

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

try {
    // Upload data.
    $result = $s3->putObject([
        'Bucket' => $bucket,
        'Key'    => $keyname,
        'Body'   => 'Hello, world!',
        'ACL'    => 'public-read'
    ]);

    // Print the URL to the object.
    echo $result['ObjectURL'] . PHP_EOL;
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Ruby

– AWS SDK for Ruby Version 3 bietet zwei Möglichkeiten, ein Objekt in Amazon S3 hochzuladen. Bei der ersten wird ein verwalteter Datei-Uploader verwendet, mit dem sich Dateien beliebiger Größe einfacher von einer Festsplatte hochladen lassen. So verwenden Sie die Methode des verwalteten Datei-Uploaders:

1. Erstellen Sie eine Instance der `Aws::S3::Resource`-Klasse.
2. Verweisen Sie per Bucket-Name und Schlüssel auf das Ziel-Objekt. Objekte befinden sich in einem Bucket und haben eindeutige Schlüssel, die jedes Objekt identifizieren.
3. Rufen Sie`#upload_file` für das Objekt auf.

Example

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectUploadFileWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Uploads a file to an Amazon S3 object by using a managed uploader.
  #
  # @param file_path [String] The path to the file to upload.
  # @return [Boolean] True when the file is uploaded; otherwise false.
  def upload_file(file_path)
    @object.upload_file(file_path)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't upload file #{file_path} to #{@object.key}. Here's why:
 #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-uploaded-file"
  file_path = "object_upload_file.rb"

  wrapper = ObjectUploadFileWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  return unless wrapper.upload_file(file_path)
```

```
  puts "File #{file_path} successfully uploaded to #{bucket_name}:#{object_key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Die zweite Möglichkeit, mit der AWS SDK for Ruby – Version 3 ein Objekt hochladen kann, ist die `#put` Methode von `Aws::S3::Object`. Das ist praktisch, wenn das Objekt eine Zeichenfolge oder ein I/O-Objekt ist, bei dem es sich nicht um eine Datei auf einem Datenträger handelt. So verwenden Sie diese Methode:

1. Erstellen Sie eine Instance der `Aws::S3::Resource`-Klasse.
2. Verweisen Sie per Bucket-Name und Schlüssel auf das Ziel-Objekt.
3. Rufen Sie `#put` auf und übergeben Sie die Zeichenfolge oder das I/O-Objekt.

Example

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectPutWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object(source_file_path)
    File.open(source_file_path, "rb") do |file|
      @object.put(body: file)
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put #{source_file_path} to #{object.key}. Here's why:
 #{e.message}"
    false
  end
end

# Example usage:
def run_demo
```

```
  bucket_name = "doc-example-bucket"
  object_key = "my-object-key"
  file_path = "my-local-file.txt"

  wrapper = ObjectPutWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  success = wrapper.put_object(file_path)
  return unless success

  puts "Put file #{file_path} into #{object_key} in #{bucket_name}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Verwenden der REST-API

Sie können REST-Anfragen senden, um ein Objekt hochzuladen. Sie können eine PUT-Anforderung senden, um Daten in einer einzigen Operation hochzuladen. Weitere Informationen finden Sie unter PUT Object.

Verwenden der AWS CLI

Sie können eine PUT-Anforderung zum Hochladen eines Objekts von bis zu 5 GB in einem einzigen Vorgang senden. Weitere Informationen finden Sie im `PutObject`-Beispiel in der AWS CLI - Befehlsreferenz.

Hochladen und Kopieren von Objekten mit mehrteiligen Uploads

Mit dem mehrteiligen Upload können Sie ein einzelnes Objekt als Satz aus mehreren Teilen hochladen. Jeder Teil ist ein zusammenhängender Teil der Daten des Objekts. Sie können diese Objektteile unabhängig und in beliebiger Reihenfolge hochladen. Wenn die Übertragung eines Teils fehlschlägt, können Sie das Teil erneut übertragen, ohne dass dies Auswirkungen auf andere Teile hat. Nachdem alle Teile Ihres Objekts hochgeladen sind, fügt Amazon S3 diese Teile zusammen und erstellt das Objekt. Wenn Ihre Objektgröße 100 MB erreicht, sollten Sie in der Regel mehrteilige Uploads verwenden, anstatt das Objekt in einem einzigen Vorgang hochzuladen.

Die Nutzung mehrteiliger Uploads bietet die folgenden Vorteile:

- Verbesserter Durchsatz – Sie können die Teile parallel hochladen, um den Durchsatz zu erhöhen.

- Schnelle Wiederherstellung bei Netzwerkproblemen – Die kleinere Teilegröße minimiert die Auswirkungen eines Neustarts eines fehlgeschlagenen Uploads aufgrund eines Netzwerkfehlers.
- Anhalten und Fortsetzen von Objekt-Uploads – Sie können Objektteile mit der Zeit hochladen. Nachdem Sie einen mehrteiligen Upload initiiert haben, gibt es kein Ablaufdatum. Sie müssen den mehrteiligen Upload ausdrücklich abschließen oder abbrechen.
- Starten Sie einen Upload, bevor Sie die endgültige Objektgröße kennen – Sie können ein Objekt hochladen, während Sie es noch erstellen.

Sie sollten den mehrteiligen Upload wie folgt verwenden:

- Wenn Sie große Objekte über ein stabiles Netzwerk mit hoher Bandbreite hochladen, können Sie einen mehrteiligen Upload verwenden, um die Nutzung der verfügbaren Bandbreite zu maximieren. Hierzu laden Sie Objektteile parallel hoch, um von einer Multi-Threading-Leistung zu profitieren.
- Wenn Sie einen Upload über ein instabiles Netzwerk ausführen, können Sie einen mehrteiligen Upload verwenden, um die Resilienz in Bezug auf Netzwerkfehler durch Vermeidung von Neustarts der Uploads zu vermeiden. Wenn Sie mehrteilige Uploads verwenden, müssen Sie nur die Teile erneut hochladen, deren Upload unterbrochen wurde. Sie müssen nicht das gesamte Objekt von Anfang an neu hochladen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets. Weitere Informationen zur Verwendung mehrteiliger Uploads mit S3 Express One Zone und Verzeichnis-Buckets finden Sie unter Verwenden von mehrteiligen Uploads mit Verzeichnis-Buckets.

Mehrteiliger Upload-Prozess

Der mehrteilige Upload ist ein Prozess in drei Schritten: Sie beginnen den Upload, laden die Teile des Objekts hoch und schließen den mehrteiligen Upload ab, wenn alle Teile hochgeladen wurden. Nach dem vollständigen Eingang der Daten aus dem mehrteiligen Upload erstellt Amazon S3 das Objekt anhand der hochgeladenen Teile. Dann können Sie auf das Objekt wie auf jedes andere Objekt in Ihrem Bucket zugreifen.

Sie können alle laufenden mehrteiligen Uploads auflisten oder eine Liste der Teile anfordern, die Sie für einen bestimmten Multipart-Upload hochgeladen haben. Alle Vorgänge werden in diesem Abschnitt erklärt.

Initiieren des mehrteiligen Uploads

Wenn Sie eine Anfrage zum Initiieren eines mehrteiligen Uploads senden, gibt Amazon S3 eine Antwort mit einer Upload-ID zurück, als eindeutige Kennung für den Multipart-Upload. Sie müssen diese Upload-ID immer angeben, wenn Sie Teile hochladen, die Teile auflisten, einen Upload abschließen oder einen Upload abbrechen. Wenn Sie Metadaten bereitstellen möchten, die das hochzuladende Objekt beschreiben, müssen sie in der Anforderung auf Initiierung des mehrteiligen Uploads angegeben werden.

Teile hochladen

Beim Hochladen eines Teils müssen Sie zusätzlich zur Upload-ID eine Teilenummer angeben. Sie können jede Teilenummer zwischen 1 und 10.000 wählen. Die Teilenummer identifiziert eindeutig einen Teil und seine Position im Objekt, das Sie hochladen. Die von Ihnen gewählte Teilenummer muss nicht fortlaufend sein (möglich sind z. B. 1, 5 und 14). Wenn Sie einen neuen Teil mit derselben Teilenummer hochladen wie bereits einmal zuvor, wird der früher hochgeladene Teil überschrieben.

Wenn Sie einen Teil hochladen, gibt Amazon S3 ein Entitäts-Tag (ETag) für den Teil als Header in der Antwort zurück. Für jeden Teilupload müssen Sie die Teilenummer und den ETag-Wert notieren. Sie müssen diese Werte in die spätere Anforderung einschließen, um den mehrteiligen Upload abzuschließen. Jeder Teil hat zum Zeitpunkt des Uploads sein eigenes ETag. Sobald der mehrteilige Upload abgeschlossen ist und alle Teile konsolidiert sind, befinden sich alle Teile jedoch unter einem ETag als Prüfsumme von Prüfsummen.

> **Note**
>
> Nachdem Sie einen mehrteiligen Upload gestartet und einen oder mehrere Teile hochgeladen haben, müssen Sie den mehrteiligen Upload abschließen oder abbrechen, damit keine Gebühren für die Speicherung der hochgeladenen Teile anfallen. Erst nach dem Abschluss oder Abbruch eines mehrteiligen Uploads gibt Amazon S3 den Speicher für die Teile frei und stoppt die Berechnung von Gebühren für die Speicherung der Teile.
> Wenn Sie einen mehrteiligen Upload abbrechen, können Sie mit dieser Upload-ID keine Teile mehr hochladen. Wenn der mehrteilige Upload abgebrochen wird, während Teile hochgeladen werden, können diese Uploads auch nach dem Abbruch des Uploads erfolgreich abgeschlossen werden oder fehlschlagen. Um sicherzustellen, dass der von allen

Teilen verbrauchte Speicherplatz freigegeben wird, dürfen Sie einen mehrteiligen Upload erst dann abbrechen, wenn alle Uploads abgeschlossen wurden.

Abschließen eines mehrteiligen Uploads

Wenn Sie einen mehrteiligen Upload abschließen, erstellt Amazon S3 ein Objekt, indem die Teile in aufsteigender Reihenfolge auf Grundlage der Teilenummer verkettet werden. Wenn Sie Metadaten für das Objekt bei der Initiierung des mehrteiligen Uploads bereitgestellt haben, verknüpft Amazon S3 die Metadaten mit dem Objekt. Nach einer erfolgreich ausgeführten Abschlussanforderung sind die Teile nicht mehr vorhanden.

Ihre Anfrage auf Abschluss des mehrteiligen Uploads muss die Upload-ID und eine Liste der Teilenummern mit den entsprechenden ETag-Werten enthalten. Die Amazon-S3-Antwort enthält einen ETag, der die kombinierten Objektdaten eindeutig identifiziert. Dieses ETag ist nicht unbedingt ein MD5-Hash der Objektdaten.

Beispielaufrufe mehrteiliger Uploads

Für dieses Beispiel nehmen wir an, dass Sie einen mehrteiligen Upload für eine 100-GB-Datei generieren. In diesem Fall hätten Sie die folgenden API-Aufrufe für den gesamten Prozess. Es würde insgesamt 1 002 API-Aufrufe geben.

- Ein `CreateMultipartUpload`-Aufruf zum Starten des Prozesses
- 1 000 individuelle `UploadPart`-Aufrufe, mit denen jeweils ein Teil von 100 MB für die Gesamtgröße von 100 GB hochgeladen wird
- Ein `CompleteMultipartUpload`-Aufruf zum Beenden des Prozesses

Auflistungen mehrteiliger Uploads

Sie können alle Teile eines bestimmten Multipart-Uploads oder alle laufenden mehrteiligen Uploads auflisten. Die Operation für die Teileauflistung gibt die Teileinformationen zurück, die Sie für einen bestimmten mehrteiligen Upload hochgeladen haben. Für jeden Abruf einer Teileauflistung gibt Amazon S3 die Teileinformationen für einen angegebenen mehrteiligen Upload bis zu maximal 1.000 Teilen zurück. Wenn im Multipart-Upload mehr als 1.000 Teile vorhanden sind, müssen Sie eine Reihe von Anforderungen auf Teileauflistung senden, um alle Teile abzurufen. Beachten Sie, dass die zurückgegebene Teileauflistung keine Teile enthält, die noch nicht vollständig hochgeladen

wurden. Bei Verwendung der Operation Mehrteilige Uploads auflisten können Sie eine Liste aller mehrteiligen Uploads erhalten, die sich in Bearbeitung befinden.

Ein mehrteiliger Upload in Verarbeitung ist ein Upload, den Sie gestartet haben, der aber noch nicht abgeschlossen ist oder abgebrochen wurde. Jeder Anforderung gibt bis zu 1.000 mehrteilige Uploads zurück. Wenn mehr als 1 000 mehrteilige Uploads vorhanden sind, müssen Sie zusätzliche Anforderungen senden, um die verbleibenden mehrteiligen Uploads abzurufen. Verwenden Sie die zurückgegebene Liste nur zur Überprüfung. Verwenden Sie das Ergebnis dieser Auflistung nicht, wenn Sie eine Anforderung für den Abschluss eines mehrteiligen Uploads senden. Halten Sie sich stattdessen an Ihre eigene Liste der Teilenummern, die Sie beim Hochladen von Teilen angegeben haben, und die diesbezüglichen ETag-Werte, die Amazon S3 zurückgegeben hat.

Prüfsummen mit mehrteiligen Upload-Operationen

Wenn Sie ein Objekt auf Amazon S3 hochladen, können Sie einen Prüfsummenalgorithmus angeben, den Amazon S3 verwenden soll. Amazon S3 verwendet standardmäßig MD5, um die Datenintegrität zu überprüfen. Sie können jedoch einen zusätzlichen Prüfsummenalgorithmus angeben, der verwendet werden soll. Bei Verwendung von MD5 berechnet Amazon S3 die Prüfsumme des gesamten mehrteiligen Objekts nach Abschluss des Uploads. Diese Prüfsumme ist keine Prüfsumme des gesamten Objekts, sondern eine Prüfsumme der Prüfsummen für jeden einzelnen Teil.

Wenn Sie Amazon S3 anweisen, zusätzliche Prüfsummen zu verwenden, berechnet Amazon S3 den Prüfsummenwert für jeden Teil und speichert die Werte. Sie können die API oder das SDK verwenden, um den Prüfsummenwert für einzelne Teile abzurufen, indem Sie `GetObject` oder `HeadObject` verwenden. Wenn Sie die Prüfsummenwerte für einzelne Teile von mehrteiligen Uploads abrufen möchten, die noch in Bearbeitung sind, können Sie `ListParts` verwenden.

> **Important**
>
> Wenn Sie einen mehrteiligen Upload mit zusätzlichen Prüfsummen verwenden, müssen die mehrteiligen Teilenummern aufeinander folgende Teilenummern sein. Wenn Sie zusätzliche Prüfsummen verwenden und versuchen, eine mehrteilige Upload-Anforderung mit nicht aufeinanderfolgenden Teilenummern abzuschließen, generiert Amazon S3 einen HTTP-Fehler `500 Internal Server Error`.

Weitere Informationen zur Funktionsweise von Prüfsummen mit mehrteiligen Objekten finden Sie unterÜberprüfung der Objektintegrität.

Gleichzeitige mehrteilige Upload-Vorgänge

In einer verteilten Entwicklungsumgebung ist es für Ihre Anwendung möglich, mehrere Updates gleichzeitig für dasselbe Objekt zu initiieren. Ihre Anwendung kann möglicherweise mehrere Multipart-Uploads mit demselben Objektschlüssel initiieren. Für jeden dieser Uploads kann Ihre Anwendung Teile hochladen und eine Anfrage auf Abschluss des Uploads an Amazon S3 senden, um das Objekt zu erstellen. Wenn die Buckets die S3-Versioning aktiviert haben, wird beim Abschluss eines mehrteiligen Uploads immer eine neue Version erstellt. Bei Buckets, für die keine Versioning aktiviert ist, kann es sein, dass zwischen den Zeitpunkten der Initiierung bis zum Abschluss eines Multipart-Uploads eine andere Anforderung vorrangig ist.

> **Note**
>
> Es ist möglich, dass eine andere Anforderung zwischen dem Zeitpunkt der Initiierung und jenem des Abschlusses eines Multipart-Uploads stattgefunden hat. Wenn z. B. ein anderer Vorgang einen Schlüssel löscht, nachdem Sie einen mehrteiligen Upload mit diesem Schlüssel initiiert haben, aber bevor Sie ihn abschließen, kann die Antwort für den Abschluss des Multipart-Uploads möglicherweise eine erfolgreiche Objekterstellung anzeigen, ohne dass Sie das Objekt je zu Gesicht bekommen haben.

Mehrteiliger Upload und Preise

Nachdem Sie einen mehrteiligen Upload gestartet haben, behält Amazon S3 alle Teile bei, bis Sie den Upload abschließen oder abbrechen. Während seiner gesamten Lebensdauer werden Ihnen der gesamte Speicher, die Bandbreite und die Anforderungen für diesen mehrteiligen Upload und die zugehörigen Teile in Rechnung gestellt.

Diese Teile werden gemäß der Speicherklasse berechnet, die beim Hochladen der Teile angegeben wurde. Eine Ausnahme bilden Teile, die in S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive hochgeladen wurden. In Bearbeitung befindliche mehrteilige Teile für einen PUT in die Speicherklasse S3 Glacier Flexible Retrieval werden als S3 Glacier Flexible Retrieval Staging Storage zu den S3-Standardspeicherpreisen in Rechnung gestellt, bis der Upload abgeschlossen ist. Darüber hinaus CreateMultipartUpload UploadPart werden sowohl als auch zu S3-Standardtarifen abgerechnet. Nur die CompleteMultipartUpload Anforderung wird zum Tarif von S3 Glacier Flexible Retrieval abgerechnet. In ähnlicher Weise werden in Bearbeitung befindliche mehrteilige Teile für einen PUT in die Speicherklasse S3 Glacier Deep Archive als S3 Glacier Flexible Retrieval Staging Storage zu den S3-Standardspeichertarifen in Rechnung gestellt, bis der Upload abgeschlossen ist,

wobei nur die CompleteMultipartUpload Anforderung zu den S3-Glacier-Deep-Archive-Preisen in Rechnung gestellt wird.

Wenn Sie den mehrteiligen Upload abbrechen, löscht Amazon S3 Upload-Artefakte und alle Teile, die Sie hochgeladen haben. Die Berechnung wird eingestellt. Für das Löschen unvollständiger mehrteiliger Uploads fallen keine Gebühren für vorzeitiges Löschen an, unabhängig von der angegebenen Speicherklasse. Weitere Informationen zu Preisen finden Sie unter Amazon-S3-Preise.

> **Note**
>
> Wir empfehlen, eine Lebenszyklusregel zu konfigurieren, durch die unvollständige mehrteilige Uploads nach einer bestimmten Anzahl von Tagen mit der Aktion `AbortIncompleteMultipartUpload` gelöscht werden, um Ihre Speicherkosten gering zu halten. Weitere Informationen zum Erstellen einer Lebenszyklusregel zum Löschen unvollständiger mehrteiliger Uploads finden Sie unter Konfigurieren einer Bucket-Lebenszykluskonfiguration zum Löschen unvollständiger mehrteiliger Uploads.

API-Unterstützung für mehrteilige Uploads

Diese Bibliotheken bieten eine hohe Abstraktion, wodurch der mehrteilige Upload von Objekten sehr einfach wird. Falls in Ihrer Anwendung erforderlich, können Sie die REST-API jedoch auch direkt verwenden. In den folgenden Abschnitten der Amazon Simple Storage Service-API-Referenz wird die REST-API für mehrteilige Uploads beschrieben.

Eine exemplarische Vorgehensweise für mehrteilige Uploads, die AWS Lambda-Funktionen verwendet, finden Sie unter Hochladen großer Objekte zu Amazon S3 mit mehrteiligem Upload und Transferbeschleunigung.

- Multipart-Upload erstellen
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

AWS Command Line Interface -Unterstützung für mehrteilige Uploads

In den folgenden Themen im werden die Operationen für mehrteilige Uploads AWS Command Line Interface beschrieben.

- Initiieren eines mehrteiligen Uploads
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

AWS SDK-Unterstützung für mehrteilige Uploads

Sie können ein Objekt mit einem AWS SDKs in Teilen hochladen. Eine Liste der AWS SDKs, die von API-Aktionen unterstützt werden, finden Sie unter:

- Multipart-Upload erstellen
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

API für mehrteilige Uploads und Berechtigungen

Sie müssen über die erforderlichen Berechtigungen verfügen, um die Multipart-Upload-Vorgänge zu verwenden. Sie können Zugriffskontrolllisten (ACLs), die Bucket-Richtlinie oder die Benutzerrichtlinie verwenden, um Einzelberechtigungen für die Ausführung dieser Vorgänge zu erteilen. Die folgende Tabelle listet die erforderlichen Berechtigungen für verschiedene mehrteilige UploadVorgänge bei der Verwendung von ACLs, einer Bucket-Richtlinie oder einer Benutzerrichtlinie auf.

Action	Erforderliche Berechtigungen
Multipart-Upload erstellen	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Multipart-Upload zu erstellen. Der Bucket-Eigentümer kann anderen Prinzipalen erlauben, die Aktion `s3:PutObject` auszuführen.
Initiieren eines mehrteiligen Uploads	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Multipart-Upload zu initiieren. Der Bucket-Eigentümer kann anderen Prinzipalen erlauben, die Aktion `s3:PutObject` auszuführen.
Initiator	Containerelement, das identifiziert, wer den Multipart-Upload initiiert hat. Wenn der Initiator ein ist AWS-Konto, stellt dieses Element die gleichen Informationen wie das Eigentümerelement bereit. Wenn der Initiator ein IAM-Benutzer ist, stellt dieses Element den Benutzer-ARN und den Anzeigenamen bereit.
Upload Part	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Teil hochzuladen. Der Bucket-Eigentümer muss dem Initiator erlauben, die Aktion `s3:PutObject` für ein Objekt auszuführen, damit der Initiator einen Teil für dieses Objekt hochladen kann.
Teil hochladen (kopieren)	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Teil hochzuladen. Da Sie einen Teil eines vorhandenen Objekts hochladen, müssen Sie die Berechtigung `s3:GetObject` für das Quellobjekt besitzen. Damit der Initiator einen Teil eines Objekts hochladen kann, muss der Bucket-Besitzer dem Initiator eine Genehmigung für die Ausführung der Aktion `s3:PutObject` für das Objekt erteilen.
Abschließen eines mehrteiligen Uploads	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Multipart-Upload abzuschließen.

Action	Erforderliche Berechtigungen
	Der Bucket-Eigentümer muss dem Initiator erlauben, die Aktion `s3:PutObject` für ein Objekt auszuführen, damit der Initiator den mehrteiligen Upload für dieses Objekt abschließen kann.
Mehrteiligen Upload abbrechen	Sie müssen die Aktion `s3:AbortMultipartUpload` ausführen dürfen, um einen mehrteiligen Upload abzubrechen. Standardmäßig können der Bucket-Eigentümer und der Initiator des mehrteiligen Uploads diese Aktion als Teil von IAM- und Bucket-Richtlinien ausführen. Wenn der Initiator ein IAM-Benutzer ist, darf der dieses Benutzers diesen mehrteiligen Upload AWS-Konto ebenfalls abbrechen. Bei VPC-Endpunktrichtlinien erhält der Initiator des mehrteiligen Uploads nicht automatisch die Berechtigung zum Ausführen der `s3:AbortMultipartUpload` -Aktion Zusätzlich zu diesen Standardberechtigungen, kann der Bucket-Eigentümer anderen Prinzipalen erlauben, die Aktion `s3:AbortMultipartUpload` auszuführen. Der Bucket-Eigentümer kann jedem Prinzipal verbieten, die Aktion `s3:AbortMultipartUpload` auszuführen.
Teile auflisten	Sie müssen die Aktion `s3:ListMultipartUploadParts` ausführen können, um Teileauflistungen in einem mehrteiligen Upload anzufragen. Standardmäßig kann der Bucket-Eigentümer Teileauflistungen von Multipart-Uploads in seinem Bucket anfordern. Der Initiator des mehrteiligen Uploads hat die Berechtigung, Teileauflistungen des spezifischen Multipart-Uploads aufzulisten. Wenn der Initiator des mehrteiligen Uploads ein IAM-Benutzer ist, hat der , der diesen IAM-Benutzer AWS-Konto steuert, auch die Berechtigung, Teile dieses Uploads aufzulisten. Zusätzlich zu diesen Standardberechtigungen, kann der Bucket-Eigentümer anderen Prinzipalen erlauben, die Aktion `s3:ListMultipartUploadParts` auszuführen. Der Bucket-Eigentümer kann jedem Prinzipal auch verbieten, die Aktion `s3:ListMultipartUploadParts` auszuführen.

Action	Erforderliche Berechtigungen
List Multipart Uploads	Sie müssen die Aktion `s3:ListBucketMultipartUploads` für einen Bucket ausführen können, um eine Multipart-Uploads-Auflistung für diesen Bucket anzufragen. Zusätzlich zu diesen Standardberechtigungen, kann der Bucket-Eigentümer anderen Prinzipalen erlauben, die Aktion `s3:ListBucketMultipartUploads` für ein Bucket auszuführen.
AWS KMS Verschlüsseln und Entschlüsseln von Berechtigungen im Zusammenhang mit	Um einen mehrteiligen Upload mit Verschlüsselung unter Verwendung eines AWS Key Management Service (AWS KMS)-KMS-Schlüssels durchzuführen, muss der Anforderer über die Berechtigung für die `kms:GenerateDataKey` Aktionen `kms:Decrypt` und für den Schlüssel verfügen. Diese Berechtigungen sind erforderlich, da Amazon S3 Daten aus den verschlüsselten Teilen der Datei entschlüsseln und lesen muss, bevor es den Multipart-Upload vornehmen kann. Weitere Informationen finden Sie unter Uploading a large file to Amazon S3 with encryption using an AWS KMS key CMK (Hochladen einer großen Datei zu Amazon S3 mit Verschlüsselung über einen KMS-CMK) im AWS -Wissenscenter. Wenn sich Ihr IAM-Benutzer oder Ihre IAM-Rolle in derselben AWS-Konto wie der KMS-Schlüssel befindet, müssen Sie über diese Berechtigungen für die Schlüsselrichtlinie verfügen. Wenn Ihr IAM-Benutzer oder Ihre Rolle zu einem anderen Konto als der KMS-Schlüssel gehört, müssen Sie über die Berechtigungen sowohl für die Schlüsselrichtlinie als auch für Ihren IAM-Benutzer oder Ihre Rolle verfügen.

Weitere Informationen zur Beziehung zwischen ACL-Berechtigungen und Berechtigungen in Zugriffsrichtlinien finden Sie unter Mapping der ACL-Berechtigungen und Zugriffsrichtlinienberechtigungen. Weitere Informationen zu IAM-Benutzern, Gruppen, Rollen und bewährten Methoden finden Sie unter Identitäten (Benutzer, Gruppen und Rollen) im IAM-Benutzerhandbuch.

Themen

- Konfigurieren einer Bucket-Lebenszykluskonfiguration zum Löschen unvollständiger mehrteiliger Uploads

- Hochladen eines Objekts mit Multipart-Upload
- Hochladen eines Verzeichnisses mit der High-Level-.NET-Klasse TransferUtility
- Auflisten von mehrteiligen Uploads
- Verfolgen eines mehrteiligen Uploads
- Abbrechen eines mehrteiligen Uploads
- Kopieren eines Objekts mit Multipart-Upload
- EInschränkungen mehrteiliger Amazon-S3-Uploads

Konfigurieren einer Bucket-Lebenszykluskonfiguration zum Löschen unvollständiger mehrteiliger Uploads

Als bewährte Methode empfehlen wir Ihnen, eine Lebenszyklusregel mit der Aktion `AbortIncompleteMultipartUpload` zu konfigurieren, um Ihre Speicherkosten zu minimieren. Weitere Informationen zum Abbruch eines mehrteiligen Uploads finden Sie unter Abbrechen eines mehrteiligen Uploads.

Amazon S3 unterstützt eine Bucket-Lebenszyklusregel, mit der Sie Amazon S3 anweisen können, unvollständige mehrteilige Uploads abzubrechen, die nicht innerhalb einer bestimmten Anzahl von Tagen nach der Initiierung abgeschlossen werden. Wenn ein mehrteiliger Upload nicht innerhalb des angegebenen Zeitrahmens abgeschlossen wird, wird er für eine Abbruchoperation zugelassen. Amazon S3 bricht den mehrteiligen Upload dann ab und löscht alle diesem mehrteiligen Upload zugeordneten Teile.

Das folgende Beispiel zeigt eine Lebenszykluskonfiguration, die eine Regel mit der Aktion `AbortIncompleteMultipartUpload` spezifiziert.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
        <Prefix></Prefix>
        <Status>Enabled</Status>
        <AbortIncompleteMultipartUpload>
          <DaysAfterInitiation>7</DaysAfterInitiation>
        </AbortIncompleteMultipartUpload>
    </Rule>
</LifecycleConfiguration>
```

In dem Beispiel gibt die Regel keinen Wert für das `Prefix`-Element (Objektschlüsselnamen-Präfix) an. Daher gilt die Regel für alle Objekte im Bucket, für die Sie mehrteilige Uploads initiiert haben. Alle mehrteiligen Uploads, die initiiert wurden und nicht innerhalb von sieben Tagen abgeschlossen wurden, werden für eine Abbruchoperation zugelassen. Die Abbruchaktion hat keine Auswirkung auf abgeschlossene mehrteilige Uploads.

Weitere Informationen zur Bucket-Lebenszykluskonfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

> **Note**
>
> Wenn der mehrteilige Upload innerhalb der in der Regel angegebenen Anzahl von Tagen abgeschlossen wurde, gilt die Lebenszyklusaktion `AbortIncompleteMultipartUpload` nicht (das heißt, Amazon S3 führt keine Aktion aus). Außerdem gilt diese Aktion nicht für Objekte. Durch diese Lebenszyklus-Aktion werden keine Objekte gelöscht. Darüber hinaus fallen keine Gebühren für das vorzeitige Löschen für den S3-Lebenszyklus an, wenn Sie Teile von unvollständigen mehrteiligen Uploads entfernen.

Verwenden der S3-Konsole

Um unvollständige mehrteilige Uploads automatisch zu verwalten, können Sie die S3-Konsole verwenden, um eine Lebenszyklusrichtlinie zu erstellen und unvollständige mehrteilige Upload-Bytes aus dem Bucket nach einer bestimmten Anzahl von Tagen ablaufen zu lassen. Im folgenden Verfahren wird gezeigt, wie Sie eine Lebenszyklusregel hinzufügen, um unvollständige mehrteilige Uploads nach 7 Tagen zu löschen. Weitere Informationen über das Hinzufügen von Lebenszyklusregeln finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

So fügen Sie eine Lebenszyklusregel hinzu, um unvollständige mehrteilige Uploads, die älter als 7 Tage sind, abzubrechen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie eine Lebenszyklusregel erstellen möchten.
3. Wählen Sie den Tab Management (Verwaltung) und dann die Option Create lifecycle rule (Lebenszyklusregel erstellen).

4. Geben Sie unter Lifecycle rule name (Name der Lebenszyklusregel) einen Namen für Ihre Regel ein.

   Der Name muss innerhalb des Buckets eindeutig sein.

5. Wählen Sie den Umfang der Lebenszyklusregel:

   - Wenn Sie eine Lebenszyklusregel für alle Objekte mit einem bestimmten Präfix erstellen möchten, wählen Sie Limit the scope of this rule using one or more filters (Den Geltungsbereich dieser Regel mithilfe eines oder mehrerer Filter einschränken) aus und geben Sie das Präfix im Feld Prefix (Präfix) ein.
   - Wenn Sie diese Lebenszyklusregel auf alle Objekte im Bucket anwenden möchten, wählen Sie This rule applies to all objects in the bucket (Diese Regel gilt für alle Objekte in dem Bucket) aus und klicken Sie auf I acknowledge that this rule applies to all objects in the bucket (Ich bestätige, dass diese Regel für alle Objekte in dem Bucket gilt).

6. Wählen Sie unter Lifecycle rule actions (Lebenszyklusregelaktionen) die Option Delete expired object delete markers or incomplete multipart uploads (Abgelaufene Objektlöschmarken oder unvollständige mehrteilige Uploads löschen) aus.
7. Wählen Sie unter Delete expired object delete markers or incomplete multipart uploads (Abgelaufene Objektlöschmarken oder unvollständige mehrteilige Uploads löschen) die Option Delete incomplete multipart uploads (Unvollständige mehrteilige Uploads löschen) aus.
8. Geben Sie im Feld Number of days (Anzahl der Tage) die Anzahl der Tage ein, nach denen unvollständige mehrteilige Uploads gelöscht werden sollen (in diesem Beispiel 7 Tage).
9. Wählen Sie Regel erstellen aus.

Verwenden der AWS CLI

Der folgende `put-bucket-lifecycle-configuration`AWS Command Line Interface (AWS CLI)-Befehl fügt die Lebenszykluskonfiguration für den angegebenen Bucket hinzu. Zum Verwenden dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3api put-bucket-lifecycle-configuration  \
        --bucket DOC-EXAMPLE-BUCKET1  \
        --lifecycle-configuration filename-containing-lifecycle-configuration
```

Das folgende Beispiel zeigt, wie Sie eine Lebenszyklusregel hinzufügen, um unvollständige mehrteilige Uploads mithilfe der AWS CLI abzubrechen. Es enthält ein Beispiel für eine JSON-

Lebenszykluskonfiguration zum Abbrechen unvollständiger mehrteiliger Uploads, die älter als 7 Tage sind.

Die CLI-Befehle in diesem Beispiel können Sie verwenden, indem Sie die *`user input placeholders`* durch Ihre Informationen ersetzen.

So fügen Sie eine Lebenszyklusregel hinzu, um unvollständige mehrteilige Uploads abzubrechen

1. Richten Sie die ein AWS CLI. Anweisungen finden Sie unter Entwickeln mit Amazon S3 über die AWS CLI.
2. Speichern Sie das folgende Beispiel einer Lebenszyklus-Konfiguration in einer Datei (z. B. *`lifecycle.json`*). Die Beispielkonfiguration gibt ein leeres Präfix an und gilt daher für alle Objekte im Bucket. Sie können ein Präfix angeben, um die Konfiguration auf eine Teilmenge von Objekten zu beschränken.

```
{
    "Rules": [
        {
            "ID": "Test Rule",
            "Status": "Enabled",
            "Filter": {
                "Prefix": ""
            },
            "AbortIncompleteMultipartUpload": {
                "DaysAfterInitiation": 7
            }
        }
    ]
}
```

3. Führen Sie den folgenden CLI-Befehl aus, um die Lebenszyklus-Konfiguration in Ihrem Bucket festzulegen.

```
aws s3api put-bucket-lifecycle-configuration   \
--bucket DOC-EXAMPLE-BUCKET1  \
--lifecycle-configuration file://lifecycle.json
```

4. Um zu überprüfen, ob die Lebenszykluskonfiguration für Ihren Bucket festgelegt wurde, rufen Sie die Lebenszykluskonfiguration mit dem folgenden `get-bucket-lifecycle`-Befehl ab.

```
aws s3api get-bucket-lifecycle  \
```

```
--bucket DOC-EXAMPLE-BUCKET1
```

5. Verwenden Sie zum Löschen der Lebenszykluskonfiguration den folgenden `delete-bucket-lifecycle`-Befehl.

```
aws s3api delete-bucket-lifecycle \
--bucket DOC-EXAMPLE-BUCKET1
```

Hochladen eines Objekts mit Multipart-Upload

Sie können den Multipart-Upload verwenden, um ein einzelnes Objekt programmgesteuert auf Amazon S3 hochzuladen.

Weitere Informationen finden Sie in den folgenden Abschnitten.

Verwenden der - AWS SDKs (High-Level-API)

Das AWS SDK stellt eine High-Level-API namens bereit`TransferManager`, die mehrteilige Uploads vereinfacht. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Sie können Daten aus einer Datei oder einem Stream hochladen. Sie können auch fortschrittliche Optionen auswählen, wie beispielsweise die Teilegröße, die Sie für den mehrteiligen Upload verwenden möchten, oder die Anzahl der gleichzeitigen Threads, die Sie für den Upload der Teile verwenden möchten. Sie können auch optionale Objekteigenschaften, die Speicherklasse oder die Access Control List (ACL) festlegen. Sie verwenden die Klassen `PutObjectRequest` und `TransferManagerConfiguration`, um die fortschrittlichen Optionen festzulegen.

Wenn möglich, versucht der `TransferManager` mehrere Threads zu verwenden, um mehrere Teile eines einzigen Uploads gleichzeitig hochzuladen. Bei großen Inhalten und hoher Bandbreite kann dies den Durchsatz erheblich erhöhen.

Zusätzlich zur Datei-Upload-Funktionalität ermöglicht die Klasse `TransferManager` Ihnen, laufende mehrteilige Uploads abzubrechen. Ein Upload wird als laufend betrachtet, nachdem Sie ihn initiiert haben und bis er abgeschlossen ist oder Sie ihn abbrechen. Der `TransferManager` bricht alle laufenden mehrteiligen Uploads für einen angegebenen Bucket ab, die vor einem angegebenen Datum und einer angegebenen Uhrzeit initiiert wurden.

Wenn Sie mehrteilige Uploads unterbrechen und fortsetzen müssen, die Teilegrößen während des Uploads ändern müssen oder die Größe der Daten nicht vorab kennen, verwenden Sie die Low-

Level-PHP-API. Weitere Informationen zu mehrteiligen Uploads, einschließlich über zusätzliche Funktionalität, die von Low-Level-API-Methoden bereitgestellt wird, finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

Java

Im folgenden Beispiel wird ein Objekt mithilfe der High-Level-Java-API für mehrteilige Uploads (der `TransferManager`-Klasse) hochgeladen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;
import com.amazonaws.services.s3.transfer.Upload;

import java.io.File;

public class HighLevelMultipartUpload {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key ***";
        String filePath = "*** Path for file to upload ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();

            // TransferManager processes all transfers asynchronously,
            // so this call returns immediately.
            Upload upload = tm.upload(bucketName, keyName, new File(filePath));
```

```
            System.out.println("Object upload started");

            // Optionally, wait for the upload to finish before continuing.
            upload.waitForCompletion();
            System.out.println("Object upload complete");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Um eine Datei in einen S3-Bucket hochzuladen, verwenden Sie die Klasse `TransferUtility`. Beim Hochladen von Daten aus einer Datei müssen Sie den Schlüsselnamen des Objekts angeben. Andernfalls verwendet die API den Dateinamen für den Schlüsselnamen. Beim Hochladen von Daten aus einem Stream müssen Sie den Schlüsselnamen des Objekts angeben.

Um fortschrittliche Upload-Optionen festzulegen – beispielsweise die Größe des Teil-Uploads, die Anzahl der Threads bei gleichzeitigem Hochladen von Upload-Teilen, Metadaten, die Speicherklasse oder die ACL –, verwenden Sie die Klasse `TransferUtilityUploadRequest`.

Im folgenden C#-Beispiel wird eine Datei in mehreren Teilen in einen Amazon-S3-Bucket hochgeladen. Es veranschaulicht, wie Sie zahlreiche `TransferUtility.Upload`-Überladungen zum Hochladen einer Datei verwenden. Jeder nachfolgende Aufruf zum Hochladen ersetzt den vorherigen Upload. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version von AWS SDK for .NET und Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.IO;
using System.Threading.Tasks;
```

```
namespace Amazon.DocSamples.S3
{
    class UploadFileMPUHighLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        private const string keyName = "*** provide a name for the uploaded object
 ***";
        private const string filePath = "*** provide the full path name of the file
 to upload ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            UploadFileAsync().Wait();
        }

        private static async Task UploadFileAsync()
        {
            try
            {
                var fileTransferUtility =
                    new TransferUtility(s3Client);

                // Option 1. Upload a file. The file name is used as the object key
 name.
                await fileTransferUtility.UploadAsync(filePath, bucketName);
                Console.WriteLine("Upload 1 completed");

                // Option 2. Specify object key name explicitly.
                await fileTransferUtility.UploadAsync(filePath, bucketName,
 keyName);
                Console.WriteLine("Upload 2 completed");

                // Option 3. Upload data from a type of System.IO.Stream.
                using (var fileToUpload =
                    new FileStream(filePath, FileMode.Open, FileAccess.Read))
                {
                    await fileTransferUtility.UploadAsync(fileToUpload,
                                                bucketName, keyName);
```

```
                }
                Console.WriteLine("Upload 3 completed");

                // Option 4. Specify advanced settings.
                var fileTransferUtilityRequest = new TransferUtilityUploadRequest
                {
                    BucketName = bucketName,
                    FilePath = filePath,
                    StorageClass = S3StorageClass.StandardInfrequentAccess,
                    PartSize = 6291456, // 6 MB.
                    Key = keyName,
                    CannedACL = S3CannedACL.PublicRead
                };
                fileTransferUtilityRequest.Metadata.Add("param1", "Value1");
                fileTransferUtilityRequest.Metadata.Add("param2", "Value2");

                await fileTransferUtility.UploadAsync(fileTransferUtilityRequest);
                Console.WriteLine("Upload 4 completed");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }

        }
    }
}
```

JavaScript

Example

Laden Sie eine große Datei hoch.

```
import {
  CreateMultipartUploadCommand,
  UploadPartCommand,
  CompleteMultipartUploadCommand,
```

```
  AbortMultipartUploadCommand,
  S3Client,
} from "@aws-sdk/client-s3";

const twentyFiveMB = 25 * 1024 * 1024;

export const createString = (size = twentyFiveMB) => {
  return "x".repeat(size);
};

export const main = async () => {
  const s3Client = new S3Client({});
  const bucketName = "test-bucket";
  const key = "multipart.txt";
  const str = createString();
  const buffer = Buffer.from(str, "utf8");

  let uploadId;

  try {
    const multipartUpload = await s3Client.send(
      new CreateMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
      }),
    );

    uploadId = multipartUpload.UploadId;

    const uploadPromises = [];
    // Multipart uploads require a minimum size of 5 MB per part.
    const partSize = Math.ceil(buffer.length / 5);

    // Upload each part.
    for (let i = 0; i < 5; i++) {
      const start = i * partSize;
      const end = start + partSize;
      uploadPromises.push(
        s3Client
          .send(
            new UploadPartCommand({
              Bucket: bucketName,
              Key: key,
              UploadId: uploadId,
```

```
            Body: buffer.subarray(start, end),
            PartNumber: i + 1,
          }),
        )
        .then((d) => {
          console.log("Part", i + 1, "uploaded");
          return d;
        }),
    );
  }

  const uploadResults = await Promise.all(uploadPromises);

  return await s3Client.send(
    new CompleteMultipartUploadCommand({
      Bucket: bucketName,
      Key: key,
      UploadId: uploadId,
      MultipartUpload: {
        Parts: uploadResults.map(({ ETag }, i) => ({
          ETag,
          PartNumber: i + 1,
        })),
      },
    }),
  );

  // Verify the output by downloading the file from the Amazon Simple Storage
Service (Amazon S3) console.
  // Because the output is a 25 MB string, text editors might struggle to open the
file.
 } catch (err) {
  console.error(err);

  if (uploadId) {
    const abortCommand = new AbortMultipartUploadCommand({
      Bucket: bucketName,
      Key: key,
      UploadId: uploadId,
    });

    await s3Client.send(abortCommand);
  }
 }
```

```
};
```

Example

Laden Sie eine große Datei herunter.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { createWriteStream } from "fs";

const s3Client = new S3Client({});
const oneMB = 1024 * 1024;

export const getObjectRange = ({ bucket, key, start, end }) => {
  const command = new GetObjectCommand({
    Bucket: bucket,
    Key: key,
    Range: `bytes=${start}-${end}`,
  });

  return s3Client.send(command);
};

export const getRangeAndLength = (contentRange) => {
  const [range, length] = contentRange.split("/");
  const [start, end] = range.split("-");
  return {
    start: parseInt(start),
    end: parseInt(end),
    length: parseInt(length),
  };
};

export const isComplete = ({ end, length }) => end === length - 1;

// When downloading a large file, you might want to break it down into
// smaller pieces. Amazon S3 accepts a Range header to specify the start
// and end of the byte range to be downloaded.
const downloadInChunks = async ({ bucket, key }) => {
  const writeStream = createWriteStream(
    fileURLToPath(new URL(`./${key}`, import.meta.url))
  ).on("error", (err) => console.error(err));

  let rangeAndLength = { start: -1, end: -1, length: -1 };
```

```
  while (!isComplete(rangeAndLength)) {
    const { end } = rangeAndLength;
    const nextRange = { start: end + 1, end: end + oneMB };

    console.log(`Downloading bytes ${nextRange.start} to ${nextRange.end}`);

    const { ContentRange, Body } = await getObjectRange({
      bucket,
      key,
      ...nextRange,
    });

    writeStream.write(await Body.transformToByteArray());
    rangeAndLength = getRangeAndLength(ContentRange);
  }
};

export const main = async () => {
  await downloadInChunks({
    bucket: "my-cool-bucket",
    key: "my-cool-object.txt",
  });
};
```

Go

Example

Laden Sie ein großes Objekt hoch, indem Sie einen Upload-Manager verwenden, um die Daten in Teile zu zerlegen und sie gleichzeitig hochzuladen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3) actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}
```

```
// UploadLargeObject uses an upload manager to upload data to an object in a bucket.
```

```
// The upload manager breaks large data into parts and uploads the parts
 concurrently.
func (basics BucketBasics) UploadLargeObject(bucketName string, objectKey string,
 largeObject []byte) error {
 largeBuffer := bytes.NewReader(largeObject)
 var partMiBs int64 = 10
 uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
  u.PartSize = partMiBs * 1024 * 1024
 })
 _, err := uploader.Upload(context.TODO(), &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
  Body:   largeBuffer,
 })
 if err != nil {
  log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }

 return err
}
```

Example

Laden Sie ein großes Objekt herunter, indem Sie einen Download-Manager verwenden, um die Daten in Teilen abzurufen und sie gleichzeitig herunterzuladen.

```
// DownloadLargeObject uses a download manager to download an object from a bucket.
// The download manager gets the data in parts and writes them to a buffer until all
 of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(bucketName string, objectKey string)
 ([]byte, error) {
 var partMiBs int64 = 10
 downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader) {
  d.PartSize = partMiBs * 1024 * 1024
 })
 buffer := manager.NewWriteAtBuffer([]byte{})
 _, err := downloader.Download(context.TODO(), buffer, &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
```

```
 })
 if err != nil {
  log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return buffer.Bytes(), err
}
```

PHP

In diesem Thema wird erläutert, wie Sie die High-Level-`Aws\S3\Model\MultipartUpload\UploadBuilder`Klasse aus AWS SDK for PHP für mehrteilige Datei-Uploads verwenden. Es wird davon ausgegangen, dass Sie den Anweisungen für folgen Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen und AWS SDK for PHP ordnungsgemäß installiert ist.

Im folgenden PHP-Beispiel wird eine Datei in einen Amazon-S3-Bucket hochgeladen. Das Beispiel veranschaulicht, wie Sie Parameter für das `MultipartUploader`-Objekt festlegen.

Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen.

```
 require 'vendor/autoload.php';

use Aws\Exception\MultipartUploadException;
use Aws\S3\MultipartUploader;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Prepare the upload parameters.
$uploader = new MultipartUploader($s3, '/path/to/large/file.zip', [
    'bucket' => $bucket,
    'key'    => $keyname
]);
```

```
// Perform the upload.
try {
    $result = $uploader->upload();
    echo "Upload complete: {$result['ObjectURL']}" . PHP_EOL;
} catch (MultipartUploadException $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Python

Im folgenden Beispiel wird ein Objekt mithilfe der High-Level-Phyton-API für mehrteilige Uploads (der `TransferManager`-Klasse) hochgeladen.

```
import sys
import threading

import boto3
from boto3.s3.transfer import TransferConfig


MB = 1024 * 1024
s3 = boto3.resource("s3")


class TransferCallback:
    """
    Handle callbacks from the transfer manager.

    The transfer manager periodically calls the __call__ method throughout
    the upload and download process so that it can take action, such as
    displaying progress to the user and collecting data about the transfer.
    """

    def __init__(self, target_size):
        self._target_size = target_size
        self._total_transferred = 0
        self._lock = threading.Lock()
        self.thread_info = {}

    def __call__(self, bytes_transferred):
        """
        The callback method that is called by the transfer manager.
```

```
        Display progress during file transfer and collect per-thread transfer
        data. This method can be called by multiple threads, so shared instance
        data is protected by a thread lock.
        """
        thread = threading.current_thread()
        with self._lock:
            self._total_transferred += bytes_transferred
            if thread.ident not in self.thread_info.keys():
                self.thread_info[thread.ident] = bytes_transferred
            else:
                self.thread_info[thread.ident] += bytes_transferred

            target = self._target_size * MB
            sys.stdout.write(
                f"\r{self._total_transferred} of {target} transferred "
                f"({(self._total_transferred / target) * 100:.2f}%)."
            )
            sys.stdout.flush()


def upload_with_default_configuration(
    local_file_path, bucket_name, object_key, file_size_mb
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, using the default
    configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def upload_with_chunksize_and_meta(
    local_file_path, bucket_name, object_key, file_size_mb, metadata=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart chunk size and adding metadata to the Amazon S3 object.

    The multipart chunk size controls the size of the chunks of data that are
    sent in the request. A smaller chunk size typically results in the transfer
```

```
    manager using more threads for the upload.

    The metadata is a set of key-value pairs that are stored with the object
    in Amazon S3.
    """
    transfer_callback = TransferCallback(file_size_mb)

    config = TransferConfig(multipart_chunksize=1 * MB)
    extra_args = {"Metadata": metadata} if metadata else None
    s3.Bucket(bucket_name).upload_file(
        local_file_path,
        object_key,
        Config=config,
        ExtraArgs=extra_args,
        Callback=transfer_callback,
    )
    return transfer_callback.thread_info


def upload_with_high_threshold(local_file_path, bucket_name, object_key,
 file_size_mb):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard upload instead of
    a multipart upload.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def upload_with_sse(
    local_file_path, bucket_name, object_key, file_size_mb, sse_key=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, adding server-side
    encryption with customer-provided encryption keys to the object.
```

```
    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)
    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey": sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, ExtraArgs=extra_args,
 Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_default_configuration(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using the
    default configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_single_thread(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using a
    single thread.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(use_threads=False)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info
```

```
def download_with_high_threshold(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard download instead
    of a multipart download.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_sse(
    bucket_name, object_key, download_file_path, file_size_mb, sse_key
):
    """
    Download a file from an Amazon S3 bucket to a local folder, adding a
    customer-provided encryption key to the request.

    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)

    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey": sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, ExtraArgs=extra_args, Callback=transfer_callback
    )
    return transfer_callback.thread_info
```

Verwenden der - AWS SDKs (Low-Level-API)

Das AWS SDK stellt eine Low-Level-API bereit, die stark an die Amazon S3-REST-API für mehrteilige Uploads erinnert (siehe Hochladen und Kopieren von Objekten mit mehrteiligen Uploads. Verwenden Sie die Low-Level-API, wenn Sie mehrteilige Uploads unterbrechen und fortsetzen müssen, die Teilegrößen während des Uploads ändern müssen oder die Größe der Upload-Daten nicht vorab kennen. Wenn Sie diese Anforderungen nicht haben, verwenden Sie die High-Level-API (siehe Verwenden der - AWS SDKs (High-Level-API)).

Java

Im folgenden Beispiel wird gezeigt, wie Sie mithilfe der Low-Level-Java-Klassen eine Datei hochladen. Es führt die folgenden Schritte aus:

- Startet einen mehrteiligen Upload mit der Methode `AmazonS3Client.initiateMultipartUpload()` und übergibt ein `InitiateMultipartUploadRequest`-Objekt.
- Speichern Sie die Upload-ID, die von der Methode `AmazonS3Client.initiateMultipartUpload()` zurückgegeben wird. Sie geben diese Upload-ID bei jeder nachfolgenden mehrteiligen Upload-Operation an.
- Lädt die Teile des Objekts hoch. Für jedes Teil rufen Sie die Methode `AmazonS3Client.uploadPart()` auf. Sie stellen die Informationen für das Hochladen von Teilen in einem `UploadPartRequest`-Objekt bereit.
- Speichert für jedes Teil das ETag aus der Antwort der Methode `AmazonS3Client.uploadPart()` in einer Liste. Sie verwenden die ETag-Werte, um den mehrteiligen Upload fertigzustellen.
- Ruft die Methode `AmazonS3Client.completeMultipartUpload()` auf, um den mehrteiligen Upload fertigzustellen.

Example

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
```

```
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class LowLevelMultipartUpload {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String filePath = "*** Path to file to upload ***";

        File file = new File(filePath);
        long contentLength = file.length();
        long partSize = 5 * 1024 * 1024; // Set part size to 5 MB.

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Create a list of ETag objects. You retrieve ETags for each object
 part
            // uploaded,
            // then, after each individual part has been uploaded, pass the list of
 ETags to
            // the request to complete the upload.
            List<PartETag> partETags = new ArrayList<PartETag>();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(bucketName, keyName);
            InitiateMultipartUploadResult initResponse =
 s3Client.initiateMultipartUpload(initRequest);

            // Upload the file parts.
            long filePosition = 0;
```

```
            for (int i = 1; filePosition < contentLength; i++) {
                // Because the last part could be less than 5 MB, adjust the part
 size as
                // needed.
                partSize = Math.min(partSize, (contentLength - filePosition));

                // Create the request to upload a part.
                UploadPartRequest uploadRequest = new UploadPartRequest()
                        .withBucketName(bucketName)
                        .withKey(keyName)
                        .withUploadId(initResponse.getUploadId())
                        .withPartNumber(i)
                        .withFileOffset(filePosition)
                        .withFile(file)
                        .withPartSize(partSize);

                // Upload the part and add the response's ETag to our list.
                UploadPartResult uploadResult = s3Client.uploadPart(uploadRequest);
                partETags.add(uploadResult.getPartETag());

                filePosition += partSize;
            }

            // Complete the multipart upload.
            CompleteMultipartUploadRequest compRequest = new
 CompleteMultipartUploadRequest(bucketName, keyName,
                    initResponse.getUploadId(), partETags);
            s3Client.completeMultipartUpload(compRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Das folgende C#-Beispiel zeigt, wie Sie die Low-Level-API für AWS SDK for .NET mehrteilige Uploads verwenden, um eine Datei in einen S3-Bucket hochzuladen. Weitere Informationen über mehrteilige Uploads zu Amazon S3 finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

> **Note**
>
> Wenn Sie die AWS SDK for .NET API verwenden, um große Objekte hochzuladen, kann ein Timeout auftreten, während Daten in den Anforderungsstream geschrieben werden. Sie können mit `UploadPartRequest` ein explizites Timeout festlegen.

In dem folgenden C#-Beispiel wird eine Datei mithilfe der Low-Level-API für mehrteilige Uploads in einen S3-Bucket hochgeladen. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version von AWS SDK for .NET und Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadFileMPULowLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        private const string keyName = "*** provide a name for the uploaded object
 ***";
        private const string filePath = "*** provide the full path name of the file
 to upload ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
```

```
        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Uploading an object");
            UploadObjectAsync().Wait();
        }

        private static async Task UploadObjectAsync()
        {
            // Create list to store upload part responses.
            List<UploadPartResponse> uploadResponses = new
List<UploadPartResponse>();

            // Setup information required to initiate the multipart upload.
            InitiateMultipartUploadRequest initiateRequest = new
InitiateMultipartUploadRequest
            {
                BucketName = bucketName,
                Key = keyName
            };

            // Initiate the upload.
            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // Upload parts.
            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
                Console.WriteLine("Uploading parts");

                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                        {
                            BucketName = bucketName,
                            Key = keyName,
                            UploadId = initResponse.UploadId,
                            PartNumber = i,
                            PartSize = partSize,
```

```
                        FilePosition = filePosition,
                        FilePath = filePath
                    };

                // Track upload progress.
                uploadRequest.StreamTransferProgress +=
                    new
EventHandler<StreamTransferProgressArgs>(UploadPartProgressEventCallback);

                // Upload a part and add the response to our list.
                uploadResponses.Add(await
s3Client.UploadPartAsync(uploadRequest));

                filePosition += partSize;
            }

            // Setup to complete the upload.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    UploadId = initResponse.UploadId
                 };
            completeRequest.AddPartETags(uploadResponses);

            // Complete the upload.
            CompleteMultipartUploadResponse completeUploadResponse =
                await s3Client.CompleteMultipartUploadAsync(completeRequest);
        }
        catch (Exception exception)
        {
            Console.WriteLine("An AmazonS3Exception was thrown: { 0}",
exception.Message);

            // Abort the upload.
            AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
            {
                BucketName = bucketName,
                Key = keyName,
                UploadId = initResponse.UploadId
            };
           await s3Client.AbortMultipartUploadAsync(abortMPURequest);
```

```
            }
        }
        public static void UploadPartProgressEventCallback(object sender,
 StreamTransferProgressArgs e)
        {
            // Process event.
            Console.WriteLine("{0}/{1}", e.TransferredBytes, e.TotalBytes);
        }
    }
}
```

PHP

In diesem Thema wird gezeigt, wie Sie die Low-Level-`uploadPart`Methode aus Version 3 des verwenden, AWS SDK for PHP um eine Datei in mehreren Teilen hochzuladen. Es wird davon ausgegangen, dass Sie den Anweisungen für folgen Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen und AWS SDK for PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel lädt eine Datei mit der PHP-Low-Level-API für mehrteilige Uploads zu einem Amazon-S3-Bucket hoch. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen.

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';
$filename = '*** Path to and Name of the File to Upload ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

$result = $s3->createMultipartUpload([
    'Bucket'       => $bucket,
    'Key'          => $keyname,
    'StorageClass' => 'REDUCED_REDUNDANCY',
    'Metadata'     => [
        'param1' => 'value 1',
        'param2' => 'value 2',
```

```
        'param3' => 'value 3'
    ]
]);
$uploadId = $result['UploadId'];

// Upload the file in parts.
try {
    $file = fopen($filename, 'r');
    $partNumber = 1;
    while (!feof($file)) {
        $result = $s3->uploadPart([
            'Bucket'     => $bucket,
            'Key'        => $keyname,
            'UploadId'   => $uploadId,
            'PartNumber' => $partNumber,
            'Body'       => fread($file, 5 * 1024 * 1024),
        ]);
        $parts['Parts'][$partNumber] = [
            'PartNumber' => $partNumber,
            'ETag' => $result['ETag'],
        ];
        $partNumber++;

        echo "Uploading part $partNumber of $filename." . PHP_EOL;
    }
    fclose($file);
} catch (S3Exception $e) {
    $result = $s3->abortMultipartUpload([
        'Bucket'   => $bucket,
        'Key'      => $keyname,
        'UploadId' => $uploadId
    ]);

    echo "Upload of $filename failed." . PHP_EOL;
}

// Complete the multipart upload.
$result = $s3->completeMultipartUpload([
    'Bucket'   => $bucket,
    'Key'      => $keyname,
    'UploadId' => $uploadId,
    'MultipartUpload'    => $parts,
]);
$url = $result['Location'];
```

```
echo "Uploaded $filename to $url." . PHP_EOL;
```

Verwenden der AWS SDK for Ruby

Die AWS SDK for Ruby Version 3 unterstützt mehrteilige Amazon S3-Uploads auf zwei Arten. Bei der ersten Option können Sie verwaltete Datei-Uploads verwenden. Weitere Informationen finden Sie unter Uploading Files to Amazon S3 im AWS -Entwickler-Blog. Verwaltete Datei-Uploads sind die empfohlene Methode zum Hochladen von Dateien zu einem Bucket. Sie bieten die folgenden Vorteile:

- Verwaltet mehrteilige Uploads von Objekten über 15 MB.
- Dateien werden im Binärmodus korrekt geöffnet, um Codierungsprobleme zu vermeiden.
- Verwendet mehrere Threads zum parallelen Hochladen von Teilen großer Objekte.

Alternativ können Sie die folgenden Multipart-Upload-Client-Vorgänge direkt verwenden:

- create_multipart_upload – Initiiert einen mehrteiligen Upload und gibt eine Upload-ID zurück.
- upload_part – Lädt einen Teil eines mehrteiligen Uploads hoch.
- upload_part_copy – Lädt einen Teil durch Kopieren von Daten aus einem vorhandenen Objekt hoch, das als Datenquelle dient.
- complete_multipart_upload – Schließt einen mehrteiligen Upload durch das Zusammenfügen zuvor hochgeladener Teile ab.
- abort_multipart_upload – Bricht einen mehrteiligen Upload ab.

Weitere Informationen finden Sie unter Verwenden von AWS SDK for Ruby – Version 3.

Verwenden der REST-API

In den folgenden Abschnitten der API-Referenz für Amazon Simple Storage Service wird die REST-API für mehrteilige Uploads beschrieben.

- Initiieren eines mehrteiligen Uploads
- Upload Part
- Abschließen eines mehrteiligen Uploads
- Mehrteiligen Upload abbrechen

- Teile auflisten
- List Multipart Uploads

Verwenden der AWS CLI

In den folgenden Abschnitten in der AWS Command Line Interface (AWS CLI) werden die Operationen für mehrteilige Uploads beschrieben.

- Initiieren eines mehrteiligen Uploads
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

Sie können auch die REST API verwenden, um Ihre eigenen REST-Anforderungen zu erstellen, oder Sie können eines der AWS -SDKs verwenden. Weitere Informationen zur REST API finden Sie unter Verwenden der REST-API. Weitere Informationen zu den SDKs finden Sie unter Hochladen eines Objekts mit Multipart-Upload.

Hochladen eines Verzeichnisses mit der High-Level-.NET-Klasse TransferUtility

Mit der Klasse `TransferUtility` können Sie ein gesamtes Verzeichnis hochladen. Standardmäßig lädt die API nur die Dateien im Stammverzeichnis des angegebenen Verzeichnisses hoch. Sie können jedoch festlegen, dass die Dateien in allen Unterverzeichnissen rekursiv hochgeladen werden sollen.

Um Dateien im angegebenen Verzeichnis basierend auf Filterkriterien anzugeben, geben Sie Filterausdrücke an. Wenn Sie z. B. nur die .pdf-Dateien aus einem Verzeichnis hochladen möchten, geben Sie den Filterausdruck `"*.pdf"` ein.

Wenn Sie Dateien aus einem Verzeichnis hochladen, geben Sie nicht die Schlüsselnamen der resultierenden Objekte an. Amazon S3 setzt die Schlüsselnamen aus dem Original-Dateipfad

zusammen. Angenommen, ein Verzeichnis mit dem Namen `c:\myfolder` besitzt die folgende Verzeichnisstruktur:

Example

```
C:\myfolder
      \a.txt
      \b.pdf
      \media\
             An.mp3
```

Wenn Sie dieses Verzeichnis hochladen, verwendet Amazon S3 die folgenden Schlüsselnamen:

Example

```
a.txt
b.pdf
media/An.mp3
```

Example

Im folgenden C#-Beispiel wird ein Verzeichnis in einen Amazon-S3-Bucket hochgeladen. Es veranschaulicht, wie Sie zahlreiche `TransferUtility.UploadDirectory`-Überladungen zum Hochladen des Verzeichnisses verwenden. Jeder nachfolgende Aufruf zum Hochladen ersetzt den vorherigen Upload. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadDirMPUHighLevelAPITest
    {
        private const string existingBucketName = "*** bucket name ***";
        private const string directoryPath = @"*** directory path ***";
        // The example uploads only .txt files.
        private const string wildCard = "*.txt";
```

```
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            UploadDirAsync().Wait();
        }

        private static async Task UploadDirAsync()
        {
            try
            {
                var directoryTransferUtility =
                    new TransferUtility(s3Client);

                // 1. Upload a directory.
                await directoryTransferUtility.UploadDirectoryAsync(directoryPath,
                    existingBucketName);
                Console.WriteLine("Upload statement 1 completed");

                // 2. Upload only the .txt files from a directory
                //    and search recursively.
                await directoryTransferUtility.UploadDirectoryAsync(
                                                directoryPath,
                                                existingBucketName,
                                                wildCard,
                                                SearchOption.AllDirectories);
                Console.WriteLine("Upload statement 2 completed");

                // 3. The same as Step 2 and some optional configuration.
                //    Search recursively for .txt files to upload.
                var request = new TransferUtilityUploadDirectoryRequest
                {
                    BucketName = existingBucketName,
                    Directory = directoryPath,
                    SearchOption = SearchOption.AllDirectories,
                    SearchPattern = wildCard
                };

                await directoryTransferUtility.UploadDirectoryAsync(request);
                Console.WriteLine("Upload statement 3 completed");
            }
            catch (AmazonS3Exception e)
```

```
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an object",
 e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Unknown encountered on server. Message:'{0}' when writing an
 object", e.Message);
            }
        }
    }
}
```

Auflisten von mehrteiligen Uploads

Sie können die - AWS SDKs (Low-Level-API) verwenden, um eine Liste der laufenden mehrteiligen Uploads in Amazon S3 abzurufen.

Auflisten mehrteiliger Uploads mit dem AWS SDK (Low-Level-API)

Java

Die folgenden Aufgaben führen Sie durch die Verwendung von Java Low-Level-Klassen, um alle laufenden mehrteiligen Uploads für einen Bucket aufzulisten.

Auflistungsprozess von mehrteiligen Uploads mithilfe der Low-Level-API

1	Erstellen Sie eine Instance der `ListMultipartUploadsRequest` -Klasse und stellen Sie den Bucket-Namen bereit.
2	Führen Sie die `AmazonS3Client.listMultipartUploads` -Methode aus. Die Methode gibt eine Instance der Klasse `MultipartUploadListing` zurück, die Ihnen Informationen über die laufenden mehrteiligen Uploads bereitstellt.

Im folgenden Java-Codebeispiel werden die vorherigen Aufgaben veranschaulicht.

Example

```
ListMultipartUploadsRequest allMultpartUploadsRequest =
     new ListMultipartUploadsRequest(existingBucketName);
MultipartUploadListing multipartUploadListing =
     s3Client.listMultipartUploads(allMultpartUploadsRequest);
```

.NET

Um alle aktuell ausgeführten mehrteiligen Uploads für einen bestimmten Bucket aufzulisten, verwenden Sie die Klasse `ListMultipartUploadsRequest` der Low-Level-API von AWS SDK for .NET für mehrteilige Uploads. Die Methode `AmazonS3Client.ListMultipartUploads` gibt eine Instance der Klasse `ListMultipartUploadsResponse` zurück, die Informationen über die laufenden mehrteiligen Uploads bereitstellt.

Ein laufender mehrteiliger Upload ist ein mehrteiliger Upload, der von der Anfrage für das Initiieren mehrteiliger Uploads initiiert, aber weder abgeschlossen noch abgebrochen wurde. Weitere Informationen über mehrteilige Amazon-S3-Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Das folgende C#-Beispiel zeigt, wie Sie die verwenden, AWS SDK for .NET um alle laufenden mehrteiligen Uploads in einem Bucket aufzulisten. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version von AWS SDK for .NET und Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
ListMultipartUploadsRequest request = new ListMultipartUploadsRequest
{
  BucketName = bucketName // Bucket receiving the uploads.
};

ListMultipartUploadsResponse response = await
 AmazonS3Client.ListMultipartUploadsAsync(request);
```

PHP

In diesem Thema wird gezeigt, wie Sie die Low-Level-API-Klassen aus Version 3 des verwenden, AWS SDK for PHP um alle laufenden mehrteiligen Uploads in einem Bucket aufzulisten. Es wird davon ausgegangen, dass Sie den Anweisungen für folgen Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen und AWS SDK for PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel zeigt eine Auflistung aller in einem Bucket laufenden mehrteiligen Uploads.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Retrieve a list of the current multipart uploads.
$result = $s3->listMultipartUploads([
    'Bucket' => $bucket
]);

// Write the list of uploads to the page.
print_r($result->toArray());
```

Auflisten von mehrteiligen Uploads mit der REST-API

In den folgenden Abschnitten der Amazon Simple Storage Service Reference wird die REST-API für das Auflisten mehrteiliger Uploads beschrieben:

- ListParts- listet die hochgeladenen Teile für einen bestimmten mehrteiligen Upload auf.
- ListMultipartUploads– Listet laufende mehrteilige Uploads auf.

Auflisten von mehrteiligen Uploads mit der AWS CLI

In den folgenden Abschnitten in der werden die Vorgänge zum Auflisten mehrteiliger Uploads AWS Command Line Interface beschrieben.

- list-parts – listet die hochgeladenen Teile für einen bestimmten mehrteiligen Upload auf.
- list-multipart-uploads– Listet laufende mehrteilige Uploads auf.

Verfolgen eines mehrteiligen Uploads

Die High-Level-API für mehrteilige Uploads bietet eine Listen-Schnittstelle, `ProgressListener`, mit der der Upload-Fortschritt verfolgt werden kann, wenn ein Objekt in Amazon S3 hochgeladen wird. Fortschrittsereignisse treten periodisch auf und benachrichtigen den Listener, dass Bytes übertragen wurden.

Java

Example

```
TransferManager tm = new TransferManager(new ProfileCredentialsProvider());

PutObjectRequest request = new PutObjectRequest(
    existingBucketName, keyName, new File(filePath));

// Subscribe to the event and provide event handler.
request.setProgressListener(new ProgressListener() {
   public void progressChanged(ProgressEvent event) {
    System.out.println("Transferred bytes: " +
      event.getBytesTransfered());
            }
});
```

Example

Der folgende Java-Code lädt eine Datei hoch und verwendet den `ProgressListener`, um den Upload-Fortschritt zu verfolgen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import java.io.File;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.event.ProgressEvent;
import com.amazonaws.event.ProgressListener;
import com.amazonaws.services.s3.model.PutObjectRequest;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.Upload;

public class TrackMPUProgressUsingHighLevelAPI {
```

```
    public static void main(String[] args) throws Exception {
        String existingBucketName = "*** Provide bucket name ***";
        String keyName            = "*** Provide object key ***";
        String filePath           = "*** file to upload ***";

        TransferManager tm = new TransferManager(new ProfileCredentialsProvider());


        // For more advanced uploads, you can create a request object
        // and supply additional request parameters (ex: progress listeners,
        // canned ACLs, etc.)
        PutObjectRequest request = new PutObjectRequest(
          existingBucketName, keyName, new File(filePath));

        // You can ask the upload for its progress, or you can
        // add a ProgressListener to your request to receive notifications
        // when bytes are transferred.
        request.setGeneralProgressListener(new ProgressListener() {
   @Override
   public void progressChanged(ProgressEvent progressEvent) {
    System.out.println("Transferred bytes: " +
      progressEvent.getBytesTransferred());
   }
  });

        // TransferManager processes all transfers asynchronously,
        // so this call will return immediately.
        Upload upload = tm.upload(request);

        try {
         // You can block and wait for the upload to finish
         upload.waitForCompletion();
        } catch (AmazonClientException amazonClientException) {
         System.out.println("Unable to upload file, upload aborted.");
         amazonClientException.printStackTrace();
        }
    }
}
```

.NET

Das folgende C#-Beispiel lädt eine Datei unter Verwendung der Klasse `TransferUtility` in einen S3-Bucket hoch und verfolgt den Fortschritt des Uploads. Informationen zur Kompatibilität

des Beispiels mit einer bestimmten Version des AWS SDK for .NET und Anleitungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TrackMPUUsingHighLevelAPITest
    {
        private const string bucketName = "*** provide the bucket name ***";
        private const string keyName = "*** provide the name for the uploaded object
 ***";
        private const string filePath = " *** provide the full path name of the file
 to upload **";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;


        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            TrackMPUAsync().Wait();
        }

        private static async Task TrackMPUAsync()
        {
            try
            {
                var fileTransferUtility = new TransferUtility(s3Client);

                // Use TransferUtilityUploadRequest to configure options.
                // In this example we subscribe to an event.
                var uploadRequest =
                    new TransferUtilityUploadRequest
                    {
                        BucketName = bucketName,
```

```
                        FilePath = filePath,
                        Key = keyName
                    };

                uploadRequest.UploadProgressEvent +=
                    new EventHandler<UploadProgressArgs>
                        (uploadRequest_UploadPartProgressEvent);

                await fileTransferUtility.UploadAsync(uploadRequest);
                Console.WriteLine("Upload completed");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }

        static void uploadRequest_UploadPartProgressEvent(object sender,
 UploadProgressArgs e)
        {
            // Process event.
            Console.WriteLine("{0}/{1}", e.TransferredBytes, e.TotalBytes);
        }
    }
}
```

Abbrechen eines mehrteiligen Uploads

Nachdem Sie einen mehrteiligen Upload initiiert haben, beginnen Sie mit dem Hochladen von Teilen. Amazon S3 speichert diese Teile, aber das Objekt wird erst dann aus den Teilen erstellt, nachdem Sie alle Teile hochgeladen und eine `successful`-Anfrage gesendet haben, um den mehrteiligen Upload abzuschließen (Sie sollten sicherstellen, dass Ihre Anfrage auf Abschluss des mehrteiligen Uploads erfolgreich ist). Nach dem Empfang der Anfrage auf Abschluss des Multipart-Uploads erzeugt Amazon S3 aus den Teilen ein Objekt. Wenn die Anforderung für den Abschluss

des mehrteiligen Uploads nicht erfolgreich ist, fügt Amazon S3 die Teile nicht zusammen und erstellt daher auch kein Objekt.

Ihnen wird der gesamte Speicher in Rechnung gestellt, der mit hochgeladenen Teilen verknüpft ist. Weitere Informationen finden Sie unter Mehrteiliger Upload und Preise. Deshalb sollten Sie den mehrteiligen Upload abschließen, damit das Objekt erstellt wird, oder den Multipart-Upload abbrechen, um hochgeladene Teile zu entfernen.

Sie können einen laufenden mehrteiligen Upload in Amazon S3 mit der AWS Command Line Interface (AWS CLI), REST-API oder AWS SDKs abbrechen. Sie können einen unvollständigen mehrteiligen Upload auch mithilfe einer Bucket-Lebenszykluskonfiguration beenden.

Verwenden der - AWS SDKs (High-Level-API)

Java

Die Klasse `TransferManager` bietet die Methode `abortMultipartUploads`, um laufende Multipart-Uploads zu stoppen. Ein Upload wird als laufend betrachtet, nachdem Sie ihn initiiert haben und bis er abgeschlossen ist oder Sie ihn abbrechen. Sie müssen einen `Date`-Wert angeben, dann bricht diese API alle mehrteiligen Uploads für diesen Bucket ab, die vor dem angegebenen `Date` initiiert wurden und noch laufen.

Die folgenden Aufgaben führen Sie durch die Verwendung von High-Level Java-Klassen für das Abbrechen mehrteiliger Uploads.

Abbruchprozess von mehrteiligen Uploads mithilfe des High-Level-API

1	Erstellen Sie eine Instance der `TransferManager` -Klasse.
2	Führen Sie die `TransferManager.abortMultipartUploads` -Methode aus, indem Sie den Bucket-Namen und einen `Date`-Wert übergeben.

Der folgende Java-Code bricht alle laufenden mehrteiligen Uploads ab, die vor mehr als einer Woche für einen bestimmten Bucket initiiert wurden. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import java.util.Date;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
```

```
import com.amazonaws.services.s3.transfer.TransferManager;

public class AbortMPUUsingHighLevelAPI {

    public static void main(String[] args) throws Exception {
        String existingBucketName = "*** Provide existing bucket name ***";

        TransferManager tm = new TransferManager(new ProfileCredentialsProvider());


        int sevenDays = 1000 * 60 * 60 * 24 * 7;
  Date oneWeekAgo = new Date(System.currentTimeMillis() - sevenDays);

        try {
         tm.abortMultipartUploads(existingBucketName, oneWeekAgo);
        } catch (AmazonClientException amazonClientException) {
         System.out.println("Unable to upload file, upload was aborted.");
         amazonClientException.printStackTrace();
        }
    }
}
```

> **Note**
>
> Sie können auch einen bestimmten mehrteiligen Upload abbrechen. Weitere Informationen finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

.NET

Das folgende C#-Bespiel bricht alle laufenden mehrteiligen Uploads ab, die vor mehr als einer Woche für einen bestimmten Bucket initiiert wurden. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version von AWS SDK for .NET und Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.Threading.Tasks;
```

```
namespace Amazon.DocSamples.S3
{
    class AbortMPUUsingHighLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            AbortMPUAsync().Wait();
        }

        private static async Task AbortMPUAsync()
        {
            try
            {
                var transferUtility = new TransferUtility(s3Client);

                // Abort all in-progress uploads initiated before the specified
 date.
                await transferUtility.AbortMultipartUploadsAsync(
                    bucketName, DateTime.Now.AddDays(-7));
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

> **Note**
>
> Sie können auch einen bestimmten mehrteiligen Upload abbrechen. Weitere Informationen finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

Verwenden der - AWS SDKs (Low-Level-API)

Durch Aufruf der Methode `AmazonS3.abortMultipartUpload` können Sie einen mehrteiligen Upload in Bearbeitung abbrechen. Diese Methode löscht alle Teile, die in Amazon S3 hochgeladen wurden, und gibt die Ressourcen frei. Sie müssen die Upload-ID, den Bucket-Namen und die Schlüsselnamen bereitstellen. Das folgende Java-Codebeispiel zeigt, wie ein laufender mehrteiliger Upload abgebrochen wird.

Um einen mehrteiligen Upload abzubrechen, stellen Sie die Upload-ID bereit sowie die Bucket- und Schlüsselnamen, die im Upload verwendet werden. Nachdem ein mehrteiliger Upload abgebrochen wurde, können Sie mit dieser Upload-ID keinen zusätzlichen Teile mehr hochladen. Weitere Informationen über mehrteilige Amazon-S3-Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Java

Im folgenden Java-Codebeispiel wird ein laufender mehrteiliger Upload gestoppt.

Example

```
InitiateMultipartUploadRequest initRequest =
    new InitiateMultipartUploadRequest(existingBucketName, keyName);
InitiateMultipartUploadResult initResponse =
               s3Client.initiateMultipartUpload(initRequest);

AmazonS3 s3Client = new AmazonS3Client(new ProfileCredentialsProvider());
s3Client.abortMultipartUpload(new AbortMultipartUploadRequest(
            existingBucketName, keyName, initResponse.getUploadId()));
```

> **Note**
>
> Statt einen spezifischen mehrteiligen Upload können Sie auch alle Ihre mehrteiligen Uploads abbrechen, die vor einem bestimmten Zeitpunkt initiiert wurden und immer noch laufen. Diese Bereinigungsoperation ist praktisch, um alte mehrteilige Uploads

abzubrechen, die Sie initiiert haben, aber die nicht abgeschlossen oder abgebrochen wurden. Weitere Informationen finden Sie unter Verwenden der - AWS SDKs (High-Level-API).

.NET

Das folgende C#-Beispiel veranschaulicht, wie Sie einen mehrteiligen Upload abbrechen. Ein vollständiges C#-Beispiel mit dem folgenden Code finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

```
AbortMultipartUploadRequest abortMPURequest = new AbortMultipartUploadRequest
{
    BucketName = existingBucketName,
    Key = keyName,
    UploadId = initResponse.UploadId
};
await AmazonS3Client.AbortMultipartUploadAsync(abortMPURequest);
```

Sie können auch alle laufenden mehrteiligen Uploads abbrechen, die vor einem bestimmten Zeitpunkt initiiert wurden. Diese Bereinigungsoperation ist praktisch, um mehrteilige Uploads abzubrechen, die nicht abgeschlossen oder abgebrochen wurden. Weitere Informationen finden Sie unter Verwenden der - AWS SDKs (High-Level-API).

PHP

Dieses Beispiel zeigt, wie Sie eine Klasse aus Version 3 des verwenden, AWS SDK for PHP um einen laufenden mehrteiligen Upload abzubrechen. Es wird davon ausgegangen, dass Sie den Anweisungen für folgen Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen und AWS SDK for PHP ordnungsgemäß installiert ist. Das Beispiel der Methode `abortMultipartUpload()`.

Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';
```

```
$uploadId = '*** Upload ID of upload to Abort ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Abort the multipart upload.
$s3->abortMultipartUpload([
    'Bucket'   => $bucket,
    'Key'      => $keyname,
    'UploadId' => $uploadId,
]);
```

Verwenden der REST-API

Weitere Informationen zur Verwendung der REST-API zum Stoppen eines mehrteiligen Uploads finden Sie unter AbortMultipartUpload in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS CLI

Weitere Informationen zur Verwendung der AWS CLI zum Stoppen eines mehrteiligen Uploads finden Sie unter abort-multipart-upload in der AWS CLI -Befehlsreferenz.

Kopieren eines Objekts mit Multipart-Upload

Die Beispiele in diesem Abschnitt zeigen Ihnen, wie Sie Objekte mit mehr als 5 GB mit Hilfe des API für mehrteilige Uploads hochladen. Sie können in einer einzelnen Operation Objekte bis zu einer Größe von 5 GB hochladen. Weitere Informationen finden Sie unter Objekte kopieren.

Verwenden der AWS SDKs

Gehen Sie zum Kopieren eines Objekts mit der Low-Level-API wie folgt vor:

- Initiieren eines mehrteiligen Uploads durch Aufrufen der Methode `AmazonS3Client.initiateMultipartUpload()`.
- Speichern Sie die Upload-ID aus dem Antwortobjekt, das die Methode `AmazonS3Client.initiateMultipartUpload()` zurückgibt. Sie geben diese Upload-ID bei jeder Teiloperation mehrteiliger Uploads an.
- Kopieren Sie alle Teile. Erstellen Sie für jeden Teil, den Sie kopieren müssen, eine neue Instance der Klasse `CopyPartRequest`. Geben Sie die teilspezifischen Informationen an, darunter Quell-

und Zielbucket-Namen, Quell- und Ziel-Objektschlüssel, Upload-ID, Stelle des ersten und des letzten Byte des Teils sowie die Nummer des Teils.

- Speichern Sie die Antworten der `AmazonS3Client.copyPart()`-Methodenaufrufe. Jede Antwort enthält den Wert für `ETag` und die Teilenummer des hochgeladenen Teils. Sie benötigen diese Informationen, um den mehrteiligen Upload abzuschließen.
- Rufen Sie die Methode `AmazonS3Client.completeMultipartUpload()` auf, um die Kopieroperation abzuschließen.

Java

Example

Das folgende Beispiel veranschaulicht, wie Sie mit der Amazon-S3-Low-Level-Java-API eine mehrteilige Kopie ausführen können. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class LowLevelMultipartCopy {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String sourceBucketName = "*** Source bucket name ***";
        String sourceObjectKey = "*** Source object key ***";
        String destBucketName = "*** Target bucket name ***";
        String destObjectKey = "*** Target object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
```

```
                    .withRegion(clientRegion)
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
InitiateMultipartUploadRequest(destBucketName,
                    destObjectKey);
            InitiateMultipartUploadResult initResult =
s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest(sourceBucketName, sourceObjectKey);
            ObjectMetadata metadataResult =
s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make
sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize -
1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(sourceBucketName)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(destBucketName)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }
```

```
            // Complete the upload request to concatenate all uploaded parts and
 make the
            // copied object available.
            CompleteMultipartUploadRequest completeRequest = new
 CompleteMultipartUploadRequest(
                    destBucketName,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
}
```

.NET

Das folgende C#-Beispiel zeigt, wie Sie die verwenden, AWS SDK for .NET um ein Amazon S3-Objekt, das größer als 5 GB ist, von einem Quellspeicherort an einen anderen zu kopieren, z. B. von einem Bucket in einen anderen. Um Objekte zu kopieren, die kleiner als 5 GB sind, verwenden Sie das Kopierverfahren in einer einzigen Operation, das unter Verwenden der AWS SDKs beschrieben wird. Weitere Informationen über mehrteilige Amazon-S3-Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Dieses Beispiel zeigt, wie Sie ein Amazon S3-Objekt, das größer als 5 GB ist, mithilfe der API AWS SDK for .NET für mehrteilige Uploads von einem S3-Bucket in einen anderen kopieren. Weitere Informationen zur SDK-Kompatibilität und Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CopyObjectUsingMPUapiTest
    {
        private const string sourceBucket = "*** provide the name of the bucket with
 source object ***";
        private const string targetBucket = "*** provide the name of the bucket to
 copy the object to ***";
        private const string sourceObjectKey = "*** provide the name of object to
 copy ***";
        private const string targetObjectKey = "*** provide the name of the object
 copy ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Copying an object");
            MPUCopyObjectAsync().Wait();
        }
        private static async Task MPUCopyObjectAsync()
        {
            // Create a list to store the upload part responses.
            List<UploadPartResponse> uploadResponses = new
 List<UploadPartResponse>();
            List<CopyPartResponse> copyResponses = new List<CopyPartResponse>();

            // Setup information required to initiate the multipart upload.
```

```
            InitiateMultipartUploadRequest initiateRequest =
                new InitiateMultipartUploadRequest
                {
                    BucketName = targetBucket,
                    Key = targetObjectKey
                };

            // Initiate the upload.
            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // Save the upload ID.
            String uploadId = initResponse.UploadId;

            try
            {
                // Get the size of the object.
                GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest
                {
                    BucketName = sourceBucket,
                    Key = sourceObjectKey
                };

                GetObjectMetadataResponse metadataResponse =
                    await s3Client.GetObjectMetadataAsync(metadataRequest);
                long objectSize = metadataResponse.ContentLength; // Length in
bytes.

                // Copy the parts.
                long partSize = 5 * (long)Math.Pow(2, 20); // Part size is 5 MB.

                long bytePosition = 0;
                for (int i = 1; bytePosition < objectSize; i++)
                {
                    CopyPartRequest copyRequest = new CopyPartRequest
                    {
                        DestinationBucket = targetBucket,
                        DestinationKey = targetObjectKey,
                        SourceBucket = sourceBucket,
                        SourceKey = sourceObjectKey,
                        UploadId = uploadId,
                        FirstByte = bytePosition,
```

```
                        LastByte = bytePosition + partSize - 1 >= objectSize ?
 objectSize - 1 : bytePosition + partSize - 1,
                        PartNumber = i
                    };

                    copyResponses.Add(await s3Client.CopyPartAsync(copyRequest));

                    bytePosition += partSize;
                }

                // Set up to complete the copy.
                CompleteMultipartUploadRequest completeRequest =
                new CompleteMultipartUploadRequest
                {
                    BucketName = targetBucket,
                    Key = targetObjectKey,
                    UploadId = initResponse.UploadId
                };
                completeRequest.AddPartETags(copyResponses);

                // Complete the copy.
                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Verwenden der REST-API

In den folgenden Abschnitten der Amazon Simple Storage Service-API-Referenz wird die REST-API für mehrteilige Uploads beschrieben. Zum Kopieren eines vorhandenen Objekts verwenden Sie

die Upload Part (Copy) API. Sie geben das Quellobjekt an, indem Sie den Anfrage-Header `x-amz-copy-source` in Ihre Anfrage aufnehmen.

- Initiieren eines mehrteiligen Uploads
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

Sie können diese APIs verwenden, um Ihre eigenen REST-Anfragen zu erstellen, oder Sie können eines der von uns bereitgestellten SDKs verwenden. Weitere Informationen zur Verwendung des mehrteiligen Uploads mit der finden Sie AWS CLI unter Verwenden der AWS CLI. Weitere Informationen zu den SDKs finden Sie unter AWS SDK-Unterstützung für mehrteilige Uploads.

EInschränkungen mehrteiliger Amazon-S3-Uploads

Die folgende Tabelle enthält die Core-Spezifikationen für den mehrteiligen Upload. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Item	Spezifikation
Maximale Objektgröße	5 TiB
Maximale Anzahl von Teilen pro Upload	10.000
Teilenummern	1 bis 10.000 (inklusive)
Teilegröße	5 MiB bis 5 GiB. Es gibt keine minimale Größenbeschränkung für den letzten Teil Ihres mehrteiligen Uploads.
Maximale Anzahl der zurückgegebenen Teile bei einer Anforderung zum Auflisten der Teile	1000

Item	Spezifikation
Maximale Anzahl der zurückgeg ebenen mehrteiligen Uploads bei einer Anforderung zum Auflisten mehrteiliger Uploads	1000

Objekte kopieren

Die Kopieroperation erzeugt eine Kopie eines Objekts, das bereits in Amazon S3 gespeichert ist.

Sie können eine Kopie Ihres Objekts mit einer Größe von bis zu 5 GB in einer einzigen atomaren Operation hochladen. Um jedoch ein Objekt zu kopieren, das größer als 5 GB ist, müssen Sie die API für mehrteilige Uploads verwenden.

Mit der `CopyObject`-Operation können Sie:

- Zusätzliche Kopien von Objekten erstellen
- Objekte umbenennen, indem Sie sie kopieren und die Originalobjekte löschen
- Objekte über Amazon-S3-Standorte verschieben (z. B. us-west-1 und Europa)
- Objektmetadaten ändern

  Jedes Amazon-S3-Objekt hat Metadaten. Dies sind Name/Wert-Paare. Sie können Objekt-Metadaten beim Hochladen festlegen. Nachdem Sie das Objekt hochgeladen haben, können Sie Objekt-Metadaten nicht mehr ändern. Die einzige Methode, wie Sie Objekt-Metadaten ändern können, ist es, eine Kopie des Objekts anzulegen und die Metadaten festzulegen. Geben Sie in der Kopieroperation dasselbe Objekt als Quelle und Ziel an.

Jedes Objekt hat Metadaten. Einige davon sind Systemmetadaten, andere sind benutzerdefiniert. Benutzer steuern einige der Systemmetadaten, wie beispielsweise die Speicherklassenkonfiguration, die für das Objekt verwendet wird. Außerdem können Sie die serverseitige Verschlüsselung konfigurieren. Wenn Sie ein Objekt kopieren, werden auch die Systemmetadaten und benutzerdefinierte Metadaten kopiert. Amazon S3 setzt die vom System gesteuerten Metadaten zurück. Wenn Sie beispielsweise ein Objekt kopieren, setzt Amazon S3 das Erstellungsdatum des kopierten Objekts zurück. Sie brauchen keine dieser Werte in Ihrer Kopieranforderung festlegen.

Wenn Sie ein Objekt kopieren, wollen Sie möglicherweise einige der Metadatenwerte aktualisieren. Wenn Ihr Quellobjekt beispielsweise für die Verwendung des S3-Standardspeichers konfiguriert ist, könnten Sie entscheiden, für die Objektkopie S3 Intelligent Tiering zu verwenden. Außerdem könnten Sie entscheiden, einige der benutzerdefinierten Metadatenwerte für das Quellobjekt zu aktualisieren. Wenn Sie entschieden haben, beim Kopieren vom Benutzer des Objekts konfigurierbare Metadaten zu aktualisieren (System oder benutzerdefiniert), müssen Sie explizit alle vom Benutzer konfigurierbaren Metadaten angeben, die im Quellobjekt Ihrer Anforderung vorhanden sind, selbst wenn Sie nur einen der Metadatenwerte ändern.

Weitere Informationen zu Objekt-Metadaten erhalten Sie unter Arbeiten mit Objekt-Metadaten.

> **Note**
>
> - Beim Kopieren von Objekten über verschiedene Standorte fallen Bandbreitengebühren an.
> - Wenn das Quellobjekt in S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive archiviert ist, müssen Sie zuerst eine temporäre Kopie wiederherstellen, bevor Sie das Objekt in einen anderen Bucket kopieren können. Weitere Informationen über das Archivieren von Objekten finden Sie unter Übergang in die Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive (Objektarchivierung).
> - Die Copy-Operation für wiederhergestellte Objekte wird in der Amazon-S3-Konsole für Objekte in der Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive nicht unterstützt. Verwenden Sie für diese Art von Kopiervorgang die AWS Command Line Interface (AWS CLI), die - AWS SDKs oder die REST-API.

Amazon S3 verschlüsselt alle neuen Objekte, die in einen S3-Bucket kopiert werden, automatisch. Wenn Sie in Ihrer Kopieranforderung keine Verschlüsselungsinformationen angeben, wird die Verschlüsselungseinstellung des Zielobjekts auf die Standardverschlüsselungskonfiguration des Ziel-Buckets festgelegt. Standardmäßig verfügen alle Buckets über eine Grundebene der Verschlüsselungskonfiguration, die eine serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verwendet. Wenn der Ziel-Bucket über eine Standardverschlüsselungskonfiguration verfügt, die serverseitige Verschlüsselung mit einem AWS Key Management Service (AWS KMS)-Schlüssel (SSE-KMS) oder einem vom Kunden bereitgestellten Verschlüsselungsschlüssel (SSE-C) verwendet, verwendet Amazon S3 den entsprechenden KMS-Schlüssel oder einen vom Kunden bereitgestellten Schlüssel, um die Zielobjektkopie zu verschlüsseln. Wenn Sie beim Kopieren eines Objekts eine andere Verschlüsselungseinstellung für das Zielobjekt verwenden möchten, können Sie festlegen,

dass Amazon S3 das Zielobjekt mit einem KMS-Schlüssel, einem von Amazon S3 verwalteten Schlüssel oder einem vom Kunden bereitgestellten Schlüssel verschlüsseln soll. Wenn sich die Verschlüsselungseinstellung in Ihrer Anforderung von der Standardverschlüsselungskonfiguration des Ziel-Buckets unterscheidet, hat die Verschlüsselungseinstellung in Ihrer Anfrage Vorrang. Wenn das Quellobjekt für den Kopiervorgang mithilfe von SSE-C in Amazon S3 gespeichert wird, müssen Sie erforderlichen Verschlüsselungsinformationen in Ihrer Anforderung bereitstellen, damit Amazon S3 das Objekt zum Kopieren entschlüsseln kann. Weitere Informationen finden Sie unter Datenschutz durch Verschlüsselung.

Beim Kopieren von Objekten können Sie einen anderen Prüfsummenalgorithmus für das Objekt verwenden. Unabhängig davon, ob Sie denselben oder einen neuen Algorithmus verwenden, berechnet Amazon S3 einen neuen Prüfsummenwert, nachdem das Objekt kopiert wurde. Amazon S3 kopiert den Wert der Prüfsumme nicht direkt. Der Prüfsummenwert von Objekten, die mit mehrteiligen Uploads geladen wurden, kann sich ändern. Weitere Informationen zum Berechnen der Prüfsumme finden Sie unter Verwenden von Prüfsummen auf Teilebene für mehrteilige Uploads.

Um mehr als ein Amazon-S3-Objekt mit einer einzigen Anforderung zu kopieren, können Sie Amazon-S3-Batch-Vorgänge verwenden. Sie stellen S3 Batch Operations eine Liste von Objekten zur Verfügung, für die Vorgänge ausgeführt werden sollen. S3-Batchoperationen rufen die entsprechende API-Operation auf, um die angegebene Operation auszuführen. Ein einzelner Batch-Vorgangsauftrag kann die angegebene Operation für Milliarden von Objekten ausführen, die Exabytes von Daten enthalten.

Die Funktion „S3-Batchoperationen“ verfolgt den Fortschritt, versendet Benachrichtigungen und speichert einen detaillierten Abschlussbericht zu allen Aktionen. Sie profitieren von einer vollständig verwalteten, prüfbaren und serverlosen Umgebung. Sie können S3-Batchoperationen über die Amazon S3-Konsole, AWS CLI, AWS SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter the section called “Grundlagen von BatchVorgänge”.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets. Weitere Informationen zum Kopieren eines Objekts zu einem Verzeichnis-Bucket finden Sie unter Hochladen eines Objekts zu einem Verzeichnis-Bucket.

Ein Objekt kopieren

Verwenden Sie die folgenden Methoden, um ein Objekt zu kopieren.

Verwenden der S3-Konsole

In der Amazon-S3-Konsole können Sie ein Objekt kopieren oder verschieben. Weitere Informationen finden Sie in den folgenden Verfahren.

> **Note**
>
> - Objekte, die mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) verschlüsselt wurden, können nicht mit der S3-Konsole kopiert oder verschoben werden. Um mit SSE-C verschlüsselte Objekte zu kopieren oder zu verschieben, verwenden Sie AWS CLI, AWS SDK oder die Amazon S3-REST-API.
> - Wenn Sie ein Objekt mit der Amazon-S3-Konsole kopieren, müssen Sie die Berechtigung `s3:ListAllMyBuckets` gewähren. Die Konsole benötigt diese Berechtigung, um den Kopiervorgang zu validieren.
> - Das regionsübergreifende Kopieren AWS KMS verschlüsselter Objekte wird auf der Amazon S3Konsole nicht unterstützt.

Ein Objekt kopieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Navigieren Sie zum Amazon-S3-Bucket oder -Ordner, der die Objekte enthält, die Sie kopieren möchten.
3. Markieren Sie das Kontrollkästchen links neben den Namen der Objekte, die Sie kopieren möchten.
4. Wählen Sie Actions (Aktionen) und wählen Sie aus der angezeigten Liste der Optionen die Option Copy (Kopieren) aus.

   Sie können auch aus den Optionen oben rechts die Option Copy (Kopieren) auswählen.
5. Wählen Sie den Zieltyp und das Zielkonto aus. Um den Zielpfad anzugeben, wählen Sie Browse S3 (S3 durchsuchen), navigieren Sie zum Ziel und markieren Sie das Kontrollkästchen links von dem Ziel. Wählen Sie unten rechts Choose destination (Ziel auswählen) aus.

Geben Sie alternativ den Zielpfad ein.

6. Wenn Sie die Bucket-Versionierung nicht aktiviert haben, werden Sie möglicherweise aufgefordert, zu bestätigen, dass vorhandene Objekte mit demselben Namen überschrieben werden. Wenn dies in Ordnung ist, markieren Sie das Kontrollkästchen und fahren Sie fort. Wenn Sie alle Versionen von Objekten in diesem Bucket behalten möchten, wählen Sie Enable Bucket Versioning (Bucket-Versionierung aktivieren). Sie können auch die Eigenschaften der Standard-Verschlüsselung und der S3-Objektsperre aktualisieren.
7. Wählen Sie unter Additional checksums (Weitere Prüfsummen) aus, ob Sie die Objekte mit der vorhandenen Prüfsummenfunktion kopieren oder die vorhandene Prüfsummenfunktion durch eine neue ersetzen möchten. Beim Hochladen der Objekte hatten Sie die Möglichkeit, den Prüfsummenalgorithmus anzugeben, der zur Überprüfung der Datenintegrität verwendet wurde. Beim Kopieren des Objekts haben Sie die Möglichkeit, eine neue Funktion auszuwählen. Wenn Sie ursprünglich keine weitere Prüfsumme angegeben haben, können Sie diesen Abschnitt der Kopieroptionen verwenden, um eine Summe hinzuzufügen.

   > **Note**
   >
   > Selbst wenn Sie dieselbe Prüfsummenfunktion verwenden, kann sich Ihr Prüfsummenwert ändern, wenn Sie das Objekt kopieren und dessen Größe 16 MB überschreitet. Der Prüfsummenwert kann sich aufgrund der Methode ändern, wie Prüfsummen für mehrteilige Uploads berechnet werden. Weitere Informationen dazu, wie sich die Prüfsumme beim Kopieren des Objekts ändern kann, finden Sie unter Verwenden von Prüfsummen auf Teilebene für mehrteilige Uploads.

   Um die Prüfsummenfunktion zu ändern, wählen Sie Replace with a new checksum function (Durch eine neue Prüfsummenfunktion ersetzen) aus. Wählen Sie die neue Prüfsummenfunktion aus dem Feld aus. Wenn das Objekt kopiert wird, wird die neue Prüfsumme mit dem angegebenen Algorithmus berechnet und gespeichert.
8. Wählen Sie unten rechts Copy (Kopieren) aus. Amazon S3 kopiert Ihre Objekte in den Zielordner.

So verschieben Sie Objekte:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Navigieren Sie zum Amazon-S3-Bucket oder -Ordner, der die Objekte enthält, die Sie verschieben möchten.
3. Markieren Sie das Kontrollkästchen links neben den Namen der Objekte, die Sie verschieben möchten.
4. Wählen Sie Actions (Aktionen) und wählen Sie aus der angezeigten Optionsliste Move (Verschieben).

   Oder wählen Sie Move (Verschieben) aus den Optionen oben rechts.
5. Um den Zielpfad anzugeben, wählen Sie Browse S3 (S3 durchsuchen), navigieren Sie zum Ziel und markieren Sie das Kontrollkästchen links von dem Ziel. Wählen Sie unten rechts Choose destination (Ziel auswählen) aus.

   Geben Sie alternativ den Zielpfad ein.
6. Wenn Sie die Bucket-Versionierung nicht aktiviert haben, werden Sie möglicherweise aufgefordert, zu bestätigen, dass vorhandene Objekte mit demselben Namen überschrieben werden. Wenn dies in Ordnung ist, markieren Sie das Kontrollkästchen und fahren Sie fort. Wenn Sie alle Versionen von Objekten in diesem Bucket behalten möchten, wählen Sie Enable Bucket Versioning (Bucket-Versionierung aktivieren). Sie können auch die Eigenschaften der Standard-Verschlüsselung und der Objektsperre aktualisieren.
7. Wählen Sie unten rechts Move (Verschieben) aus. Amazon S3 verschiebt Ihre Objekte in den Zielordner.

> **Note**
>
> - Diese Aktion erstellt eine Kopie aller angegebenen Objekte mit aktualisierten Einstellungen, aktualisiert das Datum der letzten Änderung am angegebenen Speicherort und fügt dem ursprünglichen Objekt eine Löschmarkierung hinzu.
> - Warten Sie beim Verschieben von Ordnern, bis die Verschiebungsaktion abgeschlossen ist, bevor Sie weitere Änderungen für die Ordner ausführen.
> - Diese Aktion aktualisiert Metadaten für Bucket-Versioning, Verschlüsselung, Objektsperre-Funktionen und archivierte Objekte.

Verwenden der AWS SDKs

Die Beispiele in diesem Abschnitt zeigen, wie Objekte mit bis zu 5 GB in einer einzigen Operation kopiert werden können. Für das Kopieren von Objekten mit mehr als 5 GB müssen Sie die API für mehrteilige Uploads verwenden. Weitere Informationen finden Sie unter Kopieren eines Objekts mit Multipart-Upload.

Java

Example

Das folgende Beispiel kopiert ein Objekt mit AWS SDK for Java in Amazon S3. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;

import java.io.IOException;

public class CopyObjectSingleOperation {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String sourceKey = "*** Source object key *** ";
        String destinationKey = "*** Destination object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Copy the object into a new object in the same bucket.
            CopyObjectRequest copyObjRequest = new CopyObjectRequest(bucketName,
 sourceKey, bucketName, destinationKey);
```

```
            s3Client.copyObject(copyObjRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Im folgenden C#-Beispiel wird die High-Level- verwendet, AWS SDK for .NET um Objekte bis zu einer Größe von 5 GB in einer einzigen Operation zu kopieren. Für Objekte mit mehr als 5 GB müssen Sie das in Kopieren eines Objekts mit Multipart-Upload beschriebene Kopierbeispiel für mehrteilige Uploads verwenden.

In diesem Beispiel wird eine Kopie eines Objekts bis zu einer Größe von 5 GB erstellt. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version des AWS SDK for .NET und Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CopyObjectTest
    {
        private const string sourceBucket = "*** provide the name of the bucket with
 source object ***";
        private const string destinationBucket = "*** provide the name of the bucket
 to copy the object to ***";
        private const string objectKey = "*** provide the name of object to copy
 ***";
```

```
        private const string destObjectKey = "*** provide the destination object key
 name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Copying an object");
            CopyingObjectAsync().Wait();
        }

        private static async Task CopyingObjectAsync()
        {
            try
            {
                CopyObjectRequest request = new CopyObjectRequest
                {
                    SourceBucket = sourceBucket,
                    SourceKey = objectKey,
                    DestinationBucket = destinationBucket,
                    DestinationKey = destObjectKey
                };
                CopyObjectResponse response = await
 s3Client.CopyObjectAsync(request);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

PHP

Dieses Thema führt Sie durch die Verwendung von Klassen aus Version 3 von , AWS SDK for PHP um ein einzelnes Objekt und mehrere Objekte in Amazon S3 aus einem Bucket in einen anderen oder innerhalb desselben Buckets zu kopieren.

Dieser Abschnitt setzt voraus, dass Sie den Anweisungen für Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen folgen und AWS SDK for PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel verdeutlicht die Verwendung der `copyObject()`-Methode zum Kopieren eines einzelnen Objekts in Amazon S3 sowie die Verwendung eines Aufrufstapels von `CopyObject` mit der `getcommand()`-Methode, um mehrere Kopien eines Objekts zu erstellen.

Objekte kopieren

1	Erstellen Sie mit dem Klassenkonstruktor `Aws\S3\S3Client` eine Instance eines Amazon-S3-Clients.
2	Um mehrere Kopien eines Objekts zu erstellen, führen Sie einen Stapel von Aufrufen der Methode getCommand () des Amazon S3-Clients aus, die von der Klasse Aws\CommandInterface geerbt wird. Sie stellen den `CopyObject` -Befehl als erstes Argument, sowie ein Array mit dem Quell-Bucket, dem Quellschlüsselnamen, dem Ziel-Bucket und dem Zielschlüsselnamen als zweites Argument bereit.

```
 require 'vendor/autoload.php';

use Aws\CommandPool;
use Aws\Exception\AwsException;
use Aws\ResultInterface;
use Aws\S3\S3Client;

$sourceBucket = '*** Your Source Bucket Name ***';
$sourceKeyname = '*** Your Source Object Key ***';
$targetBucket = '*** Your Target Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region' => 'us-east-1'
]);
```

```
// Copy an object.
$s3->copyObject([
    'Bucket' => $targetBucket,
    'Key' => "$sourceKeyname-copy",
    'CopySource' => "$sourceBucket/$sourceKeyname",
]);

// Perform a batch of CopyObject operations.
$batch = array();
for ($i = 1; $i <= 3; $i++) {
    $batch[] = $s3->getCommand('CopyObject', [
        'Bucket' => $targetBucket,
        'Key' => "{targetKeyname}-$i",
        'CopySource' => "$sourceBucket/$sourceKeyname",
    ]);
}
try {
    $results = CommandPool::batch($s3, $batch);
    foreach ($results as $result) {
        if ($result instanceof ResultInterface) {
            // Result handling here
        }
        if ($result instanceof AwsException) {
            // AwsException handling here
        }
    }
} catch (Exception $e) {
    // General error handling here
}
```

Python

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource in
 Boto3
                          that wraps object actions in a class-like structure.
        """
```

```
        self.object = s3_object
        self.key = self.object.key
```

```
    def copy(self, dest_object):
        """
        Copies the object to another bucket.

        :param dest_object: The destination object initialized with a bucket and
key.
                            This is a Boto3 Object resource.
        """
        try:
            dest_object.copy_from(
                CopySource={"Bucket": self.object.bucket_name, "Key":
self.object.key}
            )
            dest_object.wait_until_exists()
            logger.info(
                "Copied object from %s:%s to %s:%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
        except ClientError:
            logger.exception(
                "Couldn't copy object from %s/%s to %s/%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
            raise
```

Ruby

Die folgenden Aufgaben führen Sie durch die Verwendung der Ruby-Klassen für das Kopieren eines Objekts in Amazon S3 aus einem Bucket in einen anderen oder innerhalb desselben Buckets.

Objekte kopieren

1	Verwenden Sie das modularisierte Amazon S3-Gem für Version 3 von AWS SDK for Ruby, fordern Sie „aws-sdk-s3“ an und geben Sie Ihre AWS Anmeldeinformation en an. Weitere Informationen zur Bereitstellung Ihrer Anmeldeinformationen finden Sie in Anfragen unter Verwendung von Anmeldeinformationen von AWS-Konto oder von IAM-Benutzern.
2	Stellen Sie müssen die Anforderungsinformationen bereit, wie beispielsweise den Bucket-Namen, den Quellschlüsselnamen, den Ziel-Bucket-Namen und den Zielschlüssel.

Das folgende Ruby-Codebeispiel demonstriert die oben beschriebenen Aufgaben unter Verwendung der `#copy_object`-Methode zum Kopieren eines Objekts aus einem Bucket in einen anderen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                        copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket and rename it with the
 target key.
  #
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key)
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
```

```
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's why:
 #{e.message}"
  end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Kopieren eines Objekts mit der REST-API

Dieses Beispiel beschreibt, wie ein Objekt mit REST kopiert wird. Weitere Informationen über die REST-API finden Sie unter PUT Object (Copy).

Dieses Beispiel kopiert das `flotsam`-Objekt aus dem `pacific`-Bucket in das `jetsam`-Objekt des `atlantic`-Buckets, wobei seine Metadaten beibehalten werden.

```
PUT /jetsam HTTP/1.1
Host: atlantic.s3.amazonaws.com
x-amz-copy-source: /pacific/flotsam
Authorization: AWS AKIAIOSFODNN7EXAMPLE:ENoSbxYByFA0UGLZUqJN5EUnLDg=
Date: Wed, 20 Feb 2008 22:12:21 +0000
```

Die Signatur wurde aus den folgenden Informationen generiert.

```
PUT\r\n
```

```
\r\n
\r\n
Wed, 20 Feb 2008 22:12:21 +0000\r\n

x-amz-copy-source:/pacific/flotsam\r\n
/atlantic/jetsam
```

Amazon S3 gibt die folgende Antwort zurück, die das ETag des Objekts angibt, und wann es zuletzt geändert wurde.

```
HTTP/1.1 200 OK
x-amz-id-2: Vyaxt7qEbzv34BnSu5hctyyNSlHTYZFMWK4FtzO+iX8JQNyaLdTshL0KxatbaOZt
x-amz-request-id: 6B13C3C5B34AF333
Date: Wed, 20 Feb 2008 22:13:01 +0000

Content-Type: application/xml
Transfer-Encoding: chunked
Connection: close
Server: AmazonS3
<?xml version="1.0" encoding="UTF-8"?>

<CopyObjectResult>
   <LastModified>2008-02-20T22:13:01</LastModified>
   <ETag>"7e9c608af58950deeb370c98608ed097"</ETag>
</CopyObjectResult>
```

Verwenden der AWS CLI

Sie können auch die AWS Command Line Interface (AWS CLI) verwenden, um ein S3-Objekt zu erstellen. Weitere Informationen finden Sie unter copy-object in der AWS CLI -Befehlsreferenz.

Weitere Informationen zu finden AWS CLI Sie unter Was ist AWS Command Line Interface? im AWS Command Line Interface -Benutzerhandbuch.

Herunterladen von Objekten

In diesem Abschnitt wird erläutert, wie Sie Objekte aus einem S3-Bucket herunterladen. Mit Amazon S3 können Sie diese Objekte in einem oder mehreren Buckets speichern. Jedes Objekt kann eine Größe von bis zu 5 TB haben. Der Zugriff auf Amazon-S3-Objekte, die nicht archiviert sind, ist in Echtzeit möglich. Archivierte Objekte müssen jedoch wiederhergestellt werden, bevor sie

heruntergeladen werden können. Weitere Informationen zum Herunterladen von archivierten Objekten finden Sie unter the section called “Herunterladen von archivierten Objekten”.

Sie können ein einzelnes Objekt über die Amazon S3-Konsole, AWS Command Line Interface die (AWS CLI), AWS SDKs oder die Amazon S3-REST-API herunterladen. Verwenden Sie die S3-Konsole, um ein Objekt von S3 herunterzuladen, ohne Code schreiben oder Befehle ausführen zu müssen. Weitere Informationen finden Sie unter the section called “Herunterladen eines Objekts”.

Um mehrere Objekte herunterzuladen, verwenden Sie AWS CloudShell AWS CLI, oder die AWS SDKs . Weitere Informationen finden Sie unter the section called “Herunterladen mehrerer Objekte”.

Wenn Sie einen Teil eines Objekts herunterladen müssen, verwenden Sie zusätzliche Parameter mit der AWS CLI oder der REST-API, um nur die Bytes anzugeben, die Sie herunterladen möchten. Weitere Informationen finden Sie unter the section called “Herunterladen eines Teils eines Objekts”.

Wenn Sie ein Objekt herunterladen möchten, für das Sie keine Rechte haben, bitten Sie den Objekteigentümer, eine vorsignierte URL zu generieren, über die Sie das Objekt herunterladen können. Weitere Informationen finden Sie unter the section called “Herunterladen eines Objekts von einem anderen AWS-Konto”.

Wenn Sie Objekte außerhalb des - AWS Netzwerks herunterladen, fallen Datenübertragungsgebühren an. Die Datenübertragung innerhalb des - AWS Netzwerks ist innerhalb derselben kostenlos AWS-Region, Ihnen werden jedoch alle `GET` Anfragen in Rechnung gestellt. Weitere Informationen zu den Gebühren für Datenübertragungen und Datenabrufe finden Sie unter Preise für Amazon S3.

Themen

Herunterladen eines Objekts

Sie können ein Objekt mithilfe der Amazon S3-Konsole, AWS CLI der AWS SDKs oder der REST-API herunterladen.

Verwenden der S3-Konsole

In diesem Abschnitt erfahren Sie, wie Sie mit der Amazon-S3-Konsole ein Objekt aus einem S3-Bucket herunterladen.

> **Note**
>
> - Sie können jeweils nur ein Objekt herunterladen.
> - Wenn Sie über die Amazon-S3-Konsole ein Objekt herunterladen, dessen Schlüsselname mit Punkt (`.`) endet, wird der Punkt aus dem Schlüsselnamen des heruntergeladenen Objekts entfernt. Um den Punkt am Ende des Namens des heruntergeladenen Objekts beizubehalten, müssen Sie die AWS Command Line Interface (AWS CLI), AWS SDKs oder die Amazon S3-REST-API verwenden.

Ein Objekt von einem S3-Bucket herunterladen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, aus dem Sie ein Objekt herunterladen möchten.
3. Sie können ein Objekt wie folgt aus einem S3-Bucket herunterladen:
   - Aktivieren Sie das Kontrollkästchen neben dem Objekt und wählen Sie Herunterladen. Wenn Sie das Objekt in einen spezifischen Ordner herunterladen möchten, wählen Sie im Menü Aktionen die Option Herunterladen als.
   - Wenn Sie eine spezifische Version des Objekts herunterladen möchten, aktivieren Sie die Schaltfläche Versionen anzeigen (neben dem Suchfeld). Aktivieren Sie das Kontrollkästchen neben der gewünschten Version des Objekts und wählen Sie Herunterladen. Wenn Sie das Objekt in einen spezifischen Ordner herunterladen möchten, wählen Sie im Menü Aktionen die Option Herunterladen als.

Verwenden der AWS CLI

Das folgende Beispiel für `get-object` veranschaulicht, wie Sie die AWS CLI verwenden können, um ein Objekt von Amazon S3 herunterzuladen. Mit diesem Befehl wird das Objekt *`folder/`*

my_image aus dem Bucket *DOC-EXAMPLE-BUCKET1* abgerufen. Das Objekt wird in eine Datei mit dem Namen *my_downloaded_image* heruntergeladen.

```
aws s3api get-object --bucket DOC-EXAMPLE-BUCKET1 --key folder/
my_image my_downloaded_image
```

Weitere Informationen und Beispiele finden Sie unter get-object in der AWS CLI -Befehlsreferenz.

Verwenden der AWS SDKs

Beispiele für das Herunterladen eines Objekts mit den - AWS SDKs finden Sie unter Abrufen eines Objekts aus einem Amazon S3-Bucket mithilfe eines - AWS SDK.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Verwenden der REST-API

Sie können die REST-API verwenden, um Objekte aus Amazon S3 abzurufen. Weitere Informationen finden Sie unter GetObject in der API-Referenz zu Amazon Simple Storage Service.

Herunterladen mehrerer Objekte

Sie können mehrere Objekte herunterladen AWS CloudShell, indem Sie AWS CLI, die oder die AWS SDKs verwenden.

Verwenden von AWS CloudShell im AWS Management Console

AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt über die starten können AWS Management Console.

Weitere Informationen zu AWS CloudShell finden Sie unter Was ist CloudShell? im AWS CloudShell - Benutzerhandbuch.

> **Important**
>
> Mit verfügt AWS CloudShell Ihr Home-Verzeichnis über Speicher von bis zu 1GB pro AWS-Region. Daher können Sie keine Buckets mit Objekten synchronisieren, deren Gesamtwert diesen Wert überschreitet. Weitere Einschränkungen finden Sie unter Service quotas and restrictions im Benutzerhandbuch zu AWS CloudShell .

So laden Sie Objekte mit herunter AWS CloudShell

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudShell Konsole unter https://console.aws.amazon.com/cloudshell/.
2. Führen Sie den folgenden Befehl aus, um Objekte in Ihrem Bucket mit zu synchronisieren CloudShell. Der folgende Befehl synchronisiert Objekte aus dem Bucket *DOC-EXAMPLE-BUCKET1* und erstellt einen Ordner mit dem Namen *temp* in CloudShell. CloudShell synchronisiert Ihre Objekte mit diesem Ordner. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

   ```
   aws s3 sync s3://DOC-EXAMPLE-BUCKET1 ./temp
   ```

   > **Note**
   >
   > Wenn Sie einen Musterabgleich durchführen möchten, um bestimmte Objekte aus- oder einzuschließen, können Sie die Parameter `--exclude "value"` und `--include "value"` mit dem `sync`-Befehl verwenden.

3. Führen Sie den folgenden Befehl aus, um Ihre Objekte im Ordner *temp* in eine Datei mit dem Namen *temp.zip* zu komprimieren.

   ```
   zip temp.zip -r temp/
   ```

4. Wählen Sie Aktionen und anschließend Datei herunterladen aus.
5. Geben Sie den Dateinamen **`temp.zip`** ein und wählen Sie anschließend Herunterladen aus.
6. (Optional) Löschen Sie die *temp.zip* Datei und die Objekte, die mit dem *temp* Ordner in synchronisiert werden CloudShell. Mit AWS CloudShell haben Sie persistenten Speicher von bis zu 1 GB pro AWS-Region.

   Sie können den folgenden Beispielbefehl verwenden, um die `.zip`-Datei und den Ordner zu löschen. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

   ```
   rm temp.zip && rm -rf temp/
   ```

Verwenden der AWS CLI

Das folgende Beispiel zeigt, wie Sie die verwenden können AWS CLI , um alle Dateien oder Objekte unter dem angegebenen Verzeichnis oder Präfix herunterzuladen. Mit diesem Befehl werden alle Objekte aus dem Bucket *DOC-EXAMPLE-BUCKET1* in Ihr aktuelles Verzeichnis kopiert. Wenn Sie diesen Beispielbefehl verwenden möchten, verwenden Sie anstelle von *DOC-EXAMPLE-BUCKET1* den Namen Ihres Buckets.

```
aws s3 cp s3://DOC-EXAMPLE-BUCKET1 . --recursive
```

Mit dem folgenden Befehl werden alle Objekte unter dem Präfix *logs* im Bucket *DOC-EXAMPLE-BUCKET1* in Ihr aktuelles Verzeichnis heruntergeladen. Darüber hinaus werden die Parameter `--exclude` und `--include` verwendet, damit nur Objekte mit dem Suffix *.log* kopiert werden. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3 cp s3://DOC-EXAMPLE-BUCKET1/logs/ . --recursive --exclude "*" --include "*.log"
```

Weitere Informationen und Beispiele finden Sie unter cp in der AWS CLI -Befehlsreferenz.

Verwenden der AWS SDKs

Beispiele für das Herunterladen aller Objekte in einem Amazon S3-Bucket mit den - AWS SDKs finden Sie unter Herunterladen aller Objekte aus einem Amazon Simple Storage Service (Amazon S3)-Bucket in ein lokales Verzeichnis.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Herunterladen eines Teils eines Objekts

Sie können einen Teil eines Objekts über die AWS CLI oder die REST-API herunterladen. Dazu geben Sie mithilfe zusätzlicher Parameter an, welcher Teil eines Objekts heruntergeladen werden soll.

Verwenden der AWS CLI

Der folgende Beispielbefehl führt eine GET-Anforderung für einen Bytebereich im Objekt *folder/my_data* im Bucket *DOC-EXAMPLE-BUCKET1* aus. In der Anforderung muss dem Bytebereich

das Präfix `bytes=` vorangestellt werden. Das Teilobjekt wird in die Ausgabedatei mit dem Namen *`my_data_range`* heruntergeladen. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3api get-object --bucket DOC-EXAMPLE-BUCKET1 --key folder/my_data --range
 bytes=0-500 my_data_range
```

Weitere Informationen und Beispiele finden Sie unter get-object in der AWS CLI -Befehlsreferenz.

Weitere Informationen zum HTTP-Header `Range` finden Sie unter RFC 9110 auf der RFC-Editor-Website.

> **Note**
>
> Amazon S3 unterstützt nicht das Abrufen mehrerer Datenbereiche in einer einzelnen `GET`-Anforderung.

Verwenden der REST-API

Sie können die Parameter `partNumber` und `Range` in der REST-API verwenden, um Objektteile aus Amazon S3 abzurufen. Weitere Informationen finden Sie unter GetObject in der API-Referenz zu Amazon Simple Storage Service.

Herunterladen eines Objekts von einem anderen AWS-Konto

Sie können eine vorsignierte URL verwenden, um anderen Benutzern zeitlich begrenzten Zugriff auf Ihre Objekte zu gewähren, ohne Ihre Bucket-Richtlinie aktualisieren zu müssen.

Die vorsignierte URL kann in einem Browser eingegeben oder von einem Programm verwendet werden, um ein Objekt herunterzuladen. Die von der URL verwendeten Anmeldeinformationen sind die des AWS Benutzers, der die URL generiert hat. Nachdem die URL erstellt wurde, können alle Benutzer, die über die vorsignierte URL verfügen, das entsprechende Objekt herunterladen, bis die URL abläuft.

Verwenden einer vorsignierten URL in der S3-Konsole

Sie können die Amazon-S3-Konsole verwenden, um eine vorsignierte URL für ein Objekt zu generieren, indem Sie diese Schritte ausführen. Bei Verwendung der Konsole beträgt die maximale Ablaufzeit für eine vorsignierte URL 12 Stunden ab dem Zeitpunkt ihrer Erstellung.

So generieren Sie eine vorsignierte URL mit der Amazon-S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets mit dem Objekt aus, für das Sie eine vorsignierte URL haben möchten.
4. In der Liste Objekte wählen Sie das Objekt aus, für das Sie eine vorsignierte URL erstellen möchten.
5. Wählen Sie im Menü Objektaktionen die Option An vorsignierte URL freigeben aus.
6. Geben Sie an, wie lange die vorsignierte URL gültig sein soll.
7. Wählen Sie Create presigned URL (Vorsignierte URL erstellen).
8. Wenn eine Bestätigungsmeldung angezeigt wird, wird die URL automatisch in Ihre Zwischenablage kopiert. Sie sehen eine Schaltfläche zum Kopieren der vorsignierten URL, wenn Sie sie erneut kopieren müssen.
9. Um das Objekt herunterzuladen, fügen Sie die URL in einen beliebigen Browser ein. Daraufhin wird versucht, das Objekt herunterzuladen.

Weitere Informationen zu vorsignierten URLs und anderen Methoden, diese zu erstellen, finden Sie unter Arbeiten mit vorsignierten URLs.

Herunterladen von archivierten Objekten

Sie können Objekte, auf die selten zugegriffen wird, archivieren, um Ihre Speicherkosten dafür zu senken. Wenn Sie ein Objekt archivieren, wird es in einen kostengünstigen Speicher verschoben. Sie können also nicht in Echtzeit darauf zugreifen. Wenn Sie ein archiviertes Objekt herunterladen möchten, müssen Sie es zunächst wiederherstellen.

Je nach Speicherklasse können archivierte Objekte innerhalb von Minuten oder Stunden wiederhergestellt werden. Sie können ein archiviertes Objekt mithilfe der Amazon S3-Konsole, S3-Batchoperationen, der Amazon S3-REST-API, der AWS SDKs und der AWS Command Line Interface (AWS CLI) wiederherstellen.

Anweisungen finden Sie unter Wiederherstellen eines archivierten Objekts. Nachdem Sie das archivierte Objekt wiederhergestellt haben, können Sie es herunterladen.

Fehlerbehebung beim Herunterladen von Objekten

Unzureichende Berechtigungen oder falsche Bucket- oder AWS Identity and Access Management (IAM)-Benutzerrichtlinien können beim Versuch, Objekte von Amazon S3 herunterzuladen, zu Fehlern führen. Diese Probleme können häufig den Fehler Zugriff verweigert (403 Verboten) hervorrufen. Dies bedeutet, dass Amazon S3 den Zugriff auf eine Ressource nicht gewähren kann.

Weitere Informationen zu den häufigsten Ursachen für den Fehler Zugriff verweigert (403 Verboten) finden Sie unter Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3.

Überprüfung der Objektintegrität

Amazon S3 verwendet Prüfsummenwerte, um die Integrität von Daten zu verifizieren, die Sie auf Amazon S3 hochladen oder von Amazon S3 herunterladen. Darüber hinaus können Sie anfordern, dass ein weiterer Prüfsummenwert für jedes Objekt berechnet wird, das Sie in Amazon S3 speichern. Sie können einen von mehreren Prüfsummenalgorithmen auswählen, der beim Hochladen oder Kopieren Ihrer Daten verwendet werden soll. Amazon S3 berechnet mit diesem Algorithmus einen zusätzlichen Prüfsummenwert und speichert ihn als Teil der Objektmetadaten. Weitere Informationen zur Verwendung zusätzlicher Prüfsummen zur Überprüfung der Datenintegrität finden Sie im Tutorial: Überprüfen der Integrität von Daten in Amazon S3 mit zusätzlichen Prüfsummen.

Wenn Sie ein Objekt hochladen, können Sie optional eine vorberechnete Prüfsumme als Teil Ihrer Anforderung aufnehmen. Amazon S3 vergleicht die bereitgestellte Prüfsumme mit der Prüfsumme, die mit Ihrem angegebenen Algorithmus berechnet wird. Wenn die beiden Werte nicht übereinstimmen, meldet Amazon S3 einen Fehler.

Verwenden unterstützter Prüfsummenalgorithmen

Amazon S3 bietet Ihnen die Möglichkeit, den Prüfsummenalgorithmus auszuwählen, der zur Validierung Ihrer Daten beim Hoch- oder Herunterladen verwendet wird. Sie können einen der folgenden Secure Hash Algorithms (SHA)- oder Cyclic Redundancy Check (CRC)-Algorithmen zur Überprüfung der Datenintegrität auswählen:

- CRC32
- CRC32C
- SHA-1
- SHA-1

Wenn Sie ein Objekt hochladen, können Sie den Algorithmus angeben, den Sie verwenden möchten:

- Wenn Sie die verwenden AWS Management Console, wählen Sie den Prüfsummenalgorithmus aus, den Sie verwenden möchten. In diesem Fall können Sie den Prüfsummenwert des Objekts optional angeben. Wenn Amazon S3 das Objekt erhält, berechnet es die Prüfsumme mithilfe des von Ihnen angegebenen Algorithmus. Wenn die beiden Prüfsummenwerte nicht übereinstimmen, generiert Amazon S3 einen Fehler.
- Bei Verwendung eines SDK kkönnen Sie den Wert des `x-amz-sdk-checksum-algorithm`-Parameters für den Algorithmus festlegen, den Amazon S3 zur Berechnung der Prüfsumme verwenden soll. Amazon S3 berechnet den Prüfsummenwert automatisch.
- Bei Verwendung der REST-API nutzen Sie den `x-amz-sdk-checksum-algorithm`-Parameter nicht. Stattdessen verwenden Sie einen der algorithmenspezifischen Header (z. B. `x-amz-checksum-crc32`).

Weitere Informationen zum Hochladen von Objekten finden Sie unter Objekte hochladen.

Um einen dieser Prüfsummenwerte auf Objekte anzuwenden, die bereits auf Amazon S3 hochgeladen wurden, können Sie das Objekt kopieren. Wenn Sie ein Objekt kopieren, können Sie angeben, ob Sie den vorhandenen oder einen neuen Prüfsummenalgorithmus verwenden möchten. Sie können einen Prüfsummenalgorithmus angeben, wenn Sie eine unterstützte Methode zum Kopieren von Objekten verwenden, einschließlich S3-Batch-Operationen. Weitere Informationen über S3-Batch-Vorgänge finden Sie unter Ausführung umfangreicher Batch-Vorgänge für Amazon S3-Objekte durch..

> **Important**
>
> Wenn Sie einen mehrteiligen Upload mit zusätzlichen Prüfsummen verwenden, müssen die mehrteiligen Teilenummern aufeinander folgende Teilenummern sein. Wenn Sie zusätzliche Prüfsummen verwenden und versuchen, eine mehrteilige Upload-Anforderung mit nicht aufeinanderfolgenden Teilenummern abzuschließen, generiert Amazon S3 einen HTTP-Fehler `500 Internal Server Error`.

Nach dem Hochladen von Objekten können Sie den Prüfsummenwert abrufen und mit einem vorberechneten oder zuvor gespeicherten Prüfsummenwert vergleichen, der mit demselben Algorithmus berechnet wurde.

Verwenden der S3-Konsole

Weitere Informationen zur Verwendung der Konsole und zum Angeben von Prüfsummenalgorithmen, die beim Hochladen von Objekten verwendet werden, finden Sie unter Objekte hochladen und unter Tutorial: Überprüfen der Integrität von Daten in Amazon S3 mit zusätzlichen Prüfsummen.

Verwenden der AWS SDKs

Das folgende Beispiel zeigt, wie Sie mit den - AWS SDKs eine große Datei mit mehrteiligem Upload hochladen, eine große Datei herunterladen und eine mehrteilige Upload-Datei validieren können, wobei Sie alle SHA-256 für die Dateivalidierung verwenden.

Java

Example Beispiel: Hochladen, Herunterladen und Verifizieren einer großen Datei mit SHA-256

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import software.amazon.awssdk.auth.credentials.AwsCredentials;
import software.amazon.awssdk.auth.credentials.AwsCredentialsProvider;
import software.amazon.awssdk.core.ResponseInputStream;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.AbortMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.ChecksumAlgorithm;
import software.amazon.awssdk.services.s3.model.ChecksumMode;
import software.amazon.awssdk.services.s3.model.CompleteMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.CompleteMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.GetObjectAttributesRequest;
import software.amazon.awssdk.services.s3.model.GetObjectAttributesResponse;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingRequest;
import software.amazon.awssdk.services.s3.model.ObjectAttributes;
import software.amazon.awssdk.services.s3.model.PutObjectTaggingRequest;
import software.amazon.awssdk.services.s3.model.Tag;
import software.amazon.awssdk.services.s3.model.Tagging;
```

```
    import software.amazon.awssdk.services.s3.model.UploadPartRequest;
    import software.amazon.awssdk.services.s3.model.UploadPartResponse;

    import java.io.File;
    import java.io.FileInputStream;
    import java.io.FileOutputStream;
    import java.io.IOException;
    import java.io.InputStream;
    import java.io.OutputStream;
    import java.nio.ByteBuffer;
    import java.security.MessageDigest;
    import java.security.NoSuchAlgorithmException;
    import java.util.ArrayList;
    import java.util.Base64;
    import java.util.List;

    public class LargeObjectValidation {
        private static String FILE_NAME = "sample.file";
        private static String BUCKET = "sample-bucket";
        //Optional, if you want a method of storing the full multipart object
 checksum in S3.
        private static String CHECKSUM_TAG_KEYNAME = "fullObjectChecksum";
        //If you have existing full-object checksums that you need to validate
 against, you can do the full object validation on a sequential upload.
        private static String SHA256_FILE_BYTES = "htCM5g7ZNdoSw8bN/
mkgiAhXt5MFoVowVg+LE9aIQmI=";
        //Example Chunk Size - this must be greater than or equal to 5MB.
        private static int CHUNK_SIZE = 5 * 1024 * 1024;

        public static void main(String[] args) {
            S3Client s3Client = S3Client.builder()
                    .region(Region.US_EAST_1)
                    .credentialsProvider(new AwsCredentialsProvider() {
                        @Override
                        public AwsCredentials resolveCredentials() {
                            return new AwsCredentials() {
                                @Override
                                public String accessKeyId() {
                                    return Constants.ACCESS_KEY;
                                }

                                @Override
                                public String secretAccessKey() {
                                    return Constants.SECRET;
```

```
                                }
                            };
                        }
                    })
                    .build();
            uploadLargeFileBracketedByChecksum(s3Client);
            downloadLargeFileBracketedByChecksum(s3Client);
            validateExistingFileAgainstS3Checksum(s3Client);
        }

        public static void uploadLargeFileBracketedByChecksum(S3Client s3Client) {
            System.out.println("Starting uploading file validation");
            File file = new File(FILE_NAME);
            try (InputStream in = new FileInputStream(file)) {
                MessageDigest sha256 = MessageDigest.getInstance("SHA-256");
                CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
                        .bucket(BUCKET)
                        .key(FILE_NAME)
                        .checksumAlgorithm(ChecksumAlgorithm.SHA256)
                        .build();
                CreateMultipartUploadResponse createdUpload =
s3Client.createMultipartUpload(createMultipartUploadRequest);
                List<CompletedPart> completedParts = new ArrayList<CompletedPart>();
                int partNumber = 1;
                byte[] buffer = new byte[CHUNK_SIZE];
                int read = in.read(buffer);
                while (read != -1) {
                    UploadPartRequest uploadPartRequest =
UploadPartRequest.builder()

.partNumber(partNumber).uploadId(createdUpload.uploadId()).key(FILE_NAME).bucket(BUCKET).ch
                    UploadPartResponse uploadedPart =
s3Client.uploadPart(uploadPartRequest,
RequestBody.fromByteBuffer(ByteBuffer.wrap(buffer, 0, read)));
                    CompletedPart part =
CompletedPart.builder().partNumber(partNumber).checksumSHA256(uploadedPart.checksumSHA256()
                    completedParts.add(part);
                    sha256.update(buffer, 0, read);
                    read = in.read(buffer);
                    partNumber++;
                }
                String fullObjectChecksum =
Base64.getEncoder().encodeToString(sha256.digest());
```

```
                if (!fullObjectChecksum.equals(SHA256_FILE_BYTES)) {
                    //Because the SHA256 is uploaded after the part is uploaded; the
upload is bracketed and the full object can be fully validated.

s3Client.abortMultipartUpload(AbortMultipartUploadRequest.builder().bucket(BUCKET).key(FILE
                    throw new IOException("Byte mismatch between stored checksum and
upload, do not proceed with upload and cleanup");
                }
                CompletedMultipartUpload completedMultipartUpload =
CompletedMultipartUpload.builder().parts(completedParts).build();
                CompleteMultipartUploadResponse completedUploadResponse =
s3Client.completeMultipartUpload(

CompleteMultipartUploadRequest.builder().bucket(BUCKET).key(FILE_NAME).uploadId(createdUplo
                Tag checksumTag =
Tag.builder().key(CHECKSUM_TAG_KEYNAME).value(fullObjectChecksum).build();
                //Optionally, if you need the full object checksum stored with the
file; you could add it as a tag after completion.

s3Client.putObjectTagging(PutObjectTaggingRequest.builder().bucket(BUCKET).key(FILE_NAME).t
            } catch (IOException | NoSuchAlgorithmException e) {
                e.printStackTrace();
            }
            GetObjectAttributesResponse
                    objectAttributes =
s3Client.getObjectAttributes(GetObjectAttributesRequest.builder().bucket(BUCKET).key(FILE_N
                    .objectAttributes(ObjectAttributes.OBJECT_PARTS,
ObjectAttributes.CHECKSUM).build());
            System.out.println(objectAttributes.objectParts().parts());
            System.out.println(objectAttributes.checksum().checksumSHA256());
        }

        public static void downloadLargeFileBracketedByChecksum(S3Client s3Client) {
            System.out.println("Starting downloading file validation");
            File file = new File("DOWNLOADED_" + FILE_NAME);
            try (OutputStream out = new FileOutputStream(file)) {
                GetObjectAttributesResponse
                        objectAttributes =
s3Client.getObjectAttributes(GetObjectAttributesRequest.builder().bucket(BUCKET).key(FILE_N
                        .objectAttributes(ObjectAttributes.OBJECT_PARTS,
ObjectAttributes.CHECKSUM).build());
                //Optionally if you need the full object checksum, you can grab a
tag you added on the upload
```

```
                List<Tag> objectTags =
 s3Client.getObjectTagging(GetObjectTaggingRequest.builder().bucket(BUCKET).key(FILE_NAME).b
                String fullObjectChecksum = null;
                for (Tag objectTag : objectTags) {
                    if (objectTag.key().equals(CHECKSUM_TAG_KEYNAME)) {
                        fullObjectChecksum = objectTag.value();
                        break;
                    }
                }
                MessageDigest sha256FullObject =
 MessageDigest.getInstance("SHA-256");
                MessageDigest sha256ChecksumOfChecksums =
 MessageDigest.getInstance("SHA-256");

                //If you retrieve the object in parts, and set the ChecksumMode to
 enabled, the SDK will automatically validate the part checksum
                for (int partNumber = 1; partNumber <=
 objectAttributes.objectParts().totalPartsCount(); partNumber++) {
                    MessageDigest sha256Part = MessageDigest.getInstance("SHA-256");
                    ResponseInputStream<GetObjectResponse> response =
 s3Client.getObject(GetObjectRequest.builder().bucket(BUCKET).key(FILE_NAME).partNumber(part
                    GetObjectResponse getObjectResponse = response.response();
                    byte[] buffer = new byte[CHUNK_SIZE];
                    int read = response.read(buffer);
                    while (read != -1) {
                        out.write(buffer, 0, read);
                        sha256FullObject.update(buffer, 0, read);
                        sha256Part.update(buffer, 0, read);
                        read = response.read(buffer);
                    }
                    byte[] sha256PartBytes = sha256Part.digest();
                    sha256ChecksumOfChecksums.update(sha256PartBytes);
                    //Optionally, you can do an additional manual validation again
 the part checksum if needed in addition to the SDK check
                    String base64PartChecksum =
 Base64.getEncoder().encodeToString(sha256PartBytes);
                    String base64PartChecksumFromObjectAttributes =
 objectAttributes.objectParts().parts().get(partNumber - 1).checksumSHA256();
                    if (!
base64PartChecksum.equals(getObjectResponse.checksumSHA256()) || !
base64PartChecksum.equals(base64PartChecksumFromObjectAttributes)) {
                        throw new IOException("Part checksum didn't match for the
 part");
                    }
```

```
                    System.out.println(partNumber + " " + base64PartChecksum);
                }
                //Before finalizing, do the final checksum validation.
                String base64FullObject =
 Base64.getEncoder().encodeToString(sha256FullObject.digest());
                String base64ChecksumOfChecksums =
 Base64.getEncoder().encodeToString(sha256ChecksumOfChecksums.digest());
                if (fullObjectChecksum != null && !
fullObjectChecksum.equals(base64FullObject)) {
                    throw new IOException("Failed checksum validation for full
 object");
                }
                System.out.println(fullObjectChecksum);
                String base64ChecksumOfChecksumFromAttributes =
 objectAttributes.checksum().checksumSHA256();
                if (base64ChecksumOfChecksumFromAttributes != null && !
base64ChecksumOfChecksums.equals(base64ChecksumOfChecksumFromAttributes)) {
                    throw new IOException("Failed checksum validation for full
 object checksum of checksums");
                }
                System.out.println(base64ChecksumOfChecksumFromAttributes);
                out.flush();
            } catch (IOException | NoSuchAlgorithmException e) {
                //Cleanup bad file
                file.delete();
                e.printStackTrace();
            }
        }

        public static void validateExistingFileAgainstS3Checksum(S3Client s3Client)
 {
            System.out.println("Starting existing file validation");
            File file = new File("DOWNLOADED_" + FILE_NAME);
            GetObjectAttributesResponse
                    objectAttributes =
 s3Client.getObjectAttributes(GetObjectAttributesRequest.builder().bucket(BUCKET).key(FILE_N
                    .objectAttributes(ObjectAttributes.OBJECT_PARTS,
 ObjectAttributes.CHECKSUM).build());
            try (InputStream in = new FileInputStream(file)) {
                MessageDigest sha256ChecksumOfChecksums =
 MessageDigest.getInstance("SHA-256");
                MessageDigest sha256Part = MessageDigest.getInstance("SHA-256");
                byte[] buffer = new byte[CHUNK_SIZE];
                int currentPart = 0;
```

```
                int partBreak =
 objectAttributes.objectParts().parts().get(currentPart).size();
                int totalRead = 0;
                int read = in.read(buffer);
                while (read != -1) {
                    totalRead += read;
                    if (totalRead >= partBreak) {
                        int difference = totalRead - partBreak;
                        byte[] partChecksum;
                        if (totalRead != partBreak) {
                            sha256Part.update(buffer, 0, read - difference);
                            partChecksum = sha256Part.digest();
                            sha256ChecksumOfChecksums.update(partChecksum);
                            sha256Part.reset();
                            sha256Part.update(buffer, read - difference,
 difference);
                        } else {
                            sha256Part.update(buffer, 0, read);
                            partChecksum = sha256Part.digest();
                            sha256ChecksumOfChecksums.update(partChecksum);
                            sha256Part.reset();
                        }
                        String base64PartChecksum =
 Base64.getEncoder().encodeToString(partChecksum);
                        if (!
base64PartChecksum.equals(objectAttributes.objectParts().parts().get(currentPart).checksumSH
 {
                            throw new IOException("Part checksum didn't match S3");
                        }
                        currentPart++;
                        System.out.println(currentPart + " " + base64PartChecksum);
                        if (currentPart <
 objectAttributes.objectParts().totalPartsCount()) {
                            partBreak +=
 objectAttributes.objectParts().parts().get(currentPart - 1).size();
                        }
                    } else {
                        sha256Part.update(buffer, 0, read);
                    }
                    read = in.read(buffer);
                }
                if (currentPart != objectAttributes.objectParts().totalPartsCount())
 {
                    currentPart++;
```

```
                    byte[] partChecksum = sha256Part.digest();
                    sha256ChecksumOfChecksums.update(partChecksum);
                    String base64PartChecksum =
 Base64.getEncoder().encodeToString(partChecksum);
                    System.out.println(currentPart + " " + base64PartChecksum);
                }

                String base64CalculatedChecksumOfChecksums =
 Base64.getEncoder().encodeToString(sha256ChecksumOfChecksums.digest());
                System.out.println(base64CalculatedChecksumOfChecksums);
                System.out.println(objectAttributes.checksum().checksumSHA256());
                if (!
base64CalculatedChecksumOfChecksums.equals(objectAttributes.checksum().checksumSHA256()))
 {
                    throw new IOException("Full object checksum of checksums don't
 match S3");
                }

            } catch (IOException | NoSuchAlgorithmException e) {
                e.printStackTrace();
            }
        }
    }
```

Verwenden der REST-API

Sie können REST-Anfragen senden, um ein Objekt mit einem Prüfsummenwert hochzuladen, um die Integrität der Daten mit zu überprüfenPutObject. Sie können den Prüfsummenwert für Objekte auch mit GetObject oder abrufenHeadObject.

Verwenden der AWS CLI

Sie können eine PUT-Anforderung zum Hochladen eines Objekts von bis zu 5 GB in einem einzigen Vorgang senden. Weitere Informationen finden Sie unter `PutObject` in der AWS CLI -Befehlszeilenreferenz. Sie können auch `get-object` und `head-object` verwenden, um die Prüfsumme eines bereits hochgeladenen Objekts abzurufen und die Integrität der Daten zu überprüfen.

Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Amazon S3-CLI im AWS Command Line Interface -Benutzerhandbuch.

Verwenden von Content-MD5 beim Hochladen von Objekten

Eine andere Möglichkeit, die Integrität Ihres Objekts nach dem Hochladen zu überprüfen, besteht darin, beim Hochladen einen MD5-Digest des Objekts bereitzustellen. Wenn Sie den MD5-Digest für Ihr Objekt berechnen, können Sie den Digest mit dem `PUT`-Befehl unter Verwendung des `Content-MD5`-Headers angeben.

Nach dem Hochladen des Objekts berechnet Amazon S3 den MD5-Digest des Objekts und vergleicht ihn mit dem von Ihnen angegebenen Wert. Die Anforderung ist nur erfolgreich, wenn die beiden Digests übereinstimmen.

Die Bereitstellung eines MD5-Digest ist nicht erforderlich, aber Sie können damit die Integrität des Objekts im Rahmen des Upload-Prozesses überprüfen.

Verwenden von Content-MD5 und des ETag, um hochgeladene Objekte zu überprüfen

Das Entity-Tag (ETag) für ein Objekt stellt eine bestimmte Version dieses Objekts dar. Hinweis: Das ETag berücksichtigt nur Änderungen am Inhalt eines Objekts, nicht an seinen Metadaten. Wenn sich nur die Metadaten eines Objekts ändern, bleibt das ETag gleich.

Je nach Objekt kann das ETag des Objekts ein MD5-Digest der Objektdaten sein:

- Wenn ein Objekt über die Operation `PutObject`, `PostObject` oder `CopyObject` oder über die AWS Management Console erstellt wird und dieses Objekt außerdem Klartext oder durch serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt ist, verfügt dieses Objekt über ein ETag, das ein MD5-Digest seiner Objektdaten ist.
- Wenn ein Objekt durch die Operation `PutObject`, `PostObject`oder oder durch die erstellt wird AWS Management Console und dieses Objekt durch serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) oder durch serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) verschlüsselt wird`CopyObject`, hat dieses Objekt ein ETag, das kein MD5-Digest seiner Objektdaten ist.
- Wenn ein Objekt entweder von der `Multipart Upload`- oder der `Part Copy`-Operation erstellt wird, ist das ETag des Objekts unabhängig von der Verschlüsselungsmethode kein MD5-Digest. Wenn ein Objekt größer als 16 MB ist, kopiert oder lädt die AWS Management Console dieses Objekt als mehrteiligen Upload hoch. Daher handelt es sich beim ETag nicht um ein MD5-Digest.

Für Objekte, bei denen das ETag dem `Content-MD5`-Digest des Objekts entspricht, können Sie den ETag-Wert des Objekts mit einem berechneten oder zuvor gespeicherten `Content-MD5`-Digest vergleichen.

Verwenden von nachfolgenden Prüfsummen

Beim Hochladen von Objekten in Amazon S3 können Sie entweder eine vorberechnete Prüfsumme für das Objekt bereitstellen oder ein AWS SDK verwenden, um nachfolgende Prüfsummen automatisch in Ihrem Namen zu erstellen. Wenn Sie sich für eine nachfolgende Prüfsumme entscheiden, generiert Amazon S3 die Prüfsumme automatisch mithilfe des von Ihnen angegebenen Algorithmus und überprüft damit die Integrität des Objekts während des Uploads.

Um bei Verwendung eines AWS SDK eine nachfolgende Prüfsumme zu erstellen, füllen Sie den `ChecksumAlgorithm` Parameter mit Ihrem bevorzugten Algorithmus aus. Das SDK berechnet mit diesem Algorithmus die Prüfsumme für Ihr Objekt (oder Objektteile) und hängt sie automatisch an das Ende Ihrer Upload-Anforderung an. Dank dieser Funktionsweise sparen Sie Zeit, da Amazon S3 sowohl die Überprüfung als auch das Hochladen Ihrer Daten in einem einzigen Durchgang durchführt.

> **Important**
>
> Wenn Sie S3 Object Lambda verwenden, werden alle Anfragen an S3 Object Lambda mit `s3-object-lambda` anstelle von `s3` signiert. Dieses Verhalten wirkt sich auf die Signatur der nachfolgenden Prüfsummenwerte aus. Weitere Informationen zu S3 Object Lambda finden Sie unter Transformieren von Objekten mit S3 Object Lambda.

Verwenden von Prüfsummen auf Teilebene für mehrteilige Uploads

Wenn Objekte auf Amazon S3 hochgeladen werden, können sie entweder als einzelnes Objekt oder durch den mehrteiligen Upload-Prozess hochgeladen werden. Objekte, die größer als 16 MB sind und über die Konsole hochgeladen werden, werden automatisch mit mehrteiligen Uploads hochgeladen. Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Wenn ein Objekt als mehrteiliger Upload hochgeladen wird, ist das ETag für das Objekt kein MD5-Digest des gesamten Objekts. Amazon S3 berechnet den MD5-Digest jedes einzelnen Teils beim Hochladen. Die MD5-Digests werden verwendet, um das ETag für das endgültige Objekt zu

bestimmen. Amazon S3 verkettet die Bytes für die MD5-Digests und berechnet dann den MD5-Digest dieser verketteten Werte. Der letzte Schritt zum Erstellen des ETag besteht darin, dass Amazon S3 am Ende einen Bindestrich mit der Gesamtzahl der Teile hinzufügt.

Betrachten Sie beispielsweise ein Objekt, das mit einem mehrteiligen Upload hochgeladen wurde und das ETag `C9A5A6878D97B48CC965C1E41859F034-14` aufweist. In diesem Fall ist `C9A5A6878D97B48CC965C1E41859F034` der MD5-Digest aller miteinander verketteten Digests. Der Wert `-14` gibt an, dass mit dem mehrteiligen Upload dieses Objekts 14 Teile verknüpft sind.

Wenn Sie zusätzliche Prüfsummenwerte für Ihr mehrteiliges Objekt aktiviert haben, berechnet Amazon S3 die Prüfsumme für jeden einzelnen Teil mithilfe des angegebenen Prüfsummenalgorithmus. Die Prüfsumme für das abgeschlossene Objekt wird auf die gleiche Weise berechnet, wie Amazon S3 den MD5-Digest für den mehrteiligen Upload kalkuliert. Mithilfe dieser Prüfsumme können Sie die Integrität des Objekts überprüfen.

Um Informationen über das Objekt abzurufen, einschließlich der Anzahl der Teile, aus denen das gesamte Objekt besteht, können Sie die -GetObjectAttributesOperation verwenden. Mit zusätzlichen Prüfsummen können Sie auch Informationen für jeden einzelnen Teil wiederherstellen, der den Prüfsummenwert jedes Teils enthält.

Alternativ können Sie die Prüfsumme eines einzelnen Teils abrufen, indem Sie die - GetObject oder -HeadObjectOperation verwenden und eine Teilenummer oder einen Bytebereich angeben, der an einem einzelnen Teil ausgerichtet ist. Mit dieser Methode können Sie die Prüfsumme verwenden, um den einzelnen Teil zu validieren, ohne mit der Überprüfung der Datenintegrität warten zu müssen, bis alle Teile hochgeladen sind. Wenn Sie diese Methode verwenden, können Sie auch nur die einzelnen Teile anfordern, die die Integritätsprüfung nicht bestanden haben.

Aufgrund der Art und Weise, wie Amazon S3 die Prüfsumme für mehrteilige Objekte berechnet, kann sich der Prüfsummenwert für das Objekt ändern, wenn Sie es kopieren. Wenn Sie ein SDK oder die REST-API verwenden und aufrufenCopyObject, kopiert Amazon S3 jedes Objekt bis zu den Größenbeschränkungen der `CopyObject` API-Operation. Amazon S3 führt diese Kopie als einzelne Aktion aus, unabhängig davon, ob das Objekt in einer einzigen Anforderung oder im Rahmen eines mehrteiligen Uploads hochgeladen wurde. Mit einem Kopierbefehl ist die Prüfsumme des Objekts eine direkte Prüfsumme des vollständigen Objekts. Wenn das Objekt ursprünglich mit einem mehrteiligen Upload hochgeladen wurde, ändert sich der Prüfsummenwert, auch wenn die Daten unverändert bleiben.

> **Note**
>
> Objekte, die die Größenbeschränkungen der `CopyObject`-API-Operation überschreiten, müssen mehrteilige Kopierbefehle verwenden.

> **Important**
>
> Wenn Sie einige Operationen mit der ausführen AWS Management Console, verwendet Amazon S3 einen mehrteiligen Upload, wenn das Objekt größer als 16 MB ist. In diesem Fall ist die Prüfsumme keine direkte Prüfsumme des vollständigen Objekts, sondern eine Berechnung, die auf den Prüfsummenwerten jeden einzelnen Teils basiert.
> Betrachten Sie beispielsweise ein Objekt mit einer Größe von 100 MB, das Sie als einteiligen direkten Upload mit der REST-API hochgeladen haben. Die Prüfsumme ist in diesem Fall eine Prüfsumme des gesamten Objekts. Wenn Sie das Objekt später mit der Konsole umbenennen, es kopieren, die Speicherklasse ändern oder die Metadaten bearbeiten, verwendet Amazon S3 die mehrteilige Upload-Funktion, um das Objekt zu aktualisieren. Infolgedessen erstellt Amazon S3 einen neuen Prüfsummenwert für das Objekt, der basierend auf den Prüfsummenwerten der einzelnen Teile berechnet wird.
> Die obige Liste der Konsolenoperationen ist keine vollständige Liste aller möglichen Aktionen, die Sie in der ausführen können und die dazu führen AWS Management Console , dass Amazon S3 das Objekt mithilfe der Multipart-Upload-Funktion aktualisiert. Beachten Sie, dass der Prüfsummenwert möglicherweise nicht die Prüfsumme des gesamten Objekts ist, wenn Sie die Konsole für Objekte mit einer Größe über 16 MB verwenden.

Löschen von Amazon-S3-Objekten

Sie können ein oder mehrere Objekte direkt aus Amazon S3 mit der Amazon S3-Konsole, AWS SDKs , AWS Command Line Interface (AWS CLI) oder REST-API löschen. Für alle Objekte, die Sie in Ihrem S3-Bucket aufbewahren, entstehen Speicherkosten, deshalb sollten Sie Objekte löschen, die Sie nicht mehr brauchen. Wenn Sie beispielsweise Protokolldateien sammeln, sollten Sie sie unbedingt löschen, wenn sie nicht mehr benötigt werden. Sie können einen Lebenszyklusregel definieren, um Objekte wie Protokolldateien automatisch zu löschen. Weitere Informationen finden Sie unter the section called “Einrichten der Lebenszyklus-Konfiguration”.

Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon-S3-Preise.

Beim Löschen eines Objekts haben Sie die folgenden API-Optionen:

- Ein einzelnes Objekt löschen – Amazon S3 stellt die API-Operation `DELETE` (`DeleteObject`) bereit, mit der Sie ein Objekt innerhalb einer einzigen HTTP-Anforderung löschen können.
- Mehrere Objekte löschen – Amazon S3 stellt die API-Operation Multi-Object Delete (`DeleteObjects`) bereit, mit der Sie bis zu 1 000 Objekte innerhalb einer einzigen HTTP-Anforderung löschen können.

Wenn Sie Objekte aus einem Bucket löschen, der nicht versionsfähig ist, geben Sie nur den Namen des Objektschlüssels an. Wenn Sie jedoch Objekte aus einem versionsfähigen Bucket löschen, können Sie optional die Versions-ID des Objekts angeben, um eine bestimmte Version des Objekts zu löschen.

Programmgesteuertes Löschen von Objekten aus einem versionsfähigen Bucket

Wenn Ihr Bucket versionsfähig ist, kann es innerhalb des Buckets mehrere Versionen desselben Objekts geben. Bei der Arbeit mit versionsfähigen Buckets unterstützen die API-Operationen zum Löschen die folgenden Optionen:

- Eine Löschanforderung ohne Versions-ID angeben – Geben Sie nur den Schlüssel des Objekts an, nicht die Versions-ID. In diesem Fall erstellt Amazon S3 eine Löschmarkierung und gibt in der Antwort ihre Versions-ID zurück. Dadurch verschwindet Ihr Objekt aus dem Bucket. Weitere Informationen zum Objekt-Versioning und zum Konzept der Löschmarkierung finden Sie unter Verwenden der Versioning in S3-Buckets.
- Eine Löschanforderung mit Versions-ID angeben– Geben Sie sowohl den Schlüssel als auch eine Versions-ID an. In diesem Fall sind zwei Ergebnisse möglich:
  - Wenn die Versions-ID einer spezifischen Objektversion zugeordnet ist, löscht Amazon S3 die spezifische Version des Objekts.
  - Wenn die Versions-ID der Löschmarkierung dieses Objekts zugeordnet ist, löscht Amazon S3 die Löschmarkierung. Dadurch wird Ihr Objekt wieder in Ihrem Bucket angezeigt.

Löschen von Objekten aus einem MFA-fähigen Bucket

Beachten Sie beim Löschen von Objekten aus einem MFA-fähigen Bucket (Multi-Factor-Authentication) Folgendes:

- Wenn Sie ein ungültiges MFA-Token bereitstellen, schlägt die Anforderung immer fehl.
- Wenn Sie einen MFA-fähigen Bucket haben und eine versionsfähige Löschanforderung erstellen (d. h. Sie geben einen Objektschlüssel und eine Versions-ID an), schlägt die Anforderung fehl, wenn Sie kein gültiges MFA-Token bereitstellen. Wenn Sie die API-Operation Multi-Object Delete für einen MFA-fähigen Bucket verwenden und eine der Löschanforderungen versionsfähig ist (d. h. Sie geben einen Objektschlüssel und eine Versions-ID an), schlägt die gesamte Anforderung fehl, wenn Sie kein MFA-Token bereitstellen.

In den folgenden Fällen ist die Anfrage jedoch erfolgreich:

- Wenn Sie einen MFA-fähigen Bucket besitzen und eine nicht versionsfähige Löschanforderung ausführen (d. h. kein versionsfähiges Objekt löschen) und Sie kein MFA-Token bereitstellen, wird die Löschanforderung erfolgreich ausgeführt.
- Wenn Sie die Anforderung Multi-Object Delete ausführen und ausschließlich nicht versionsfähige Objekte zur Löschung aus einem MFA-fähigen Bucket angeben und Sie kein MFA-Token bereitstellen, werden die Objekte erfolgreich gelöscht.

Informationen zum Löschen von MFAs finden Sie unter Konfigurieren von MFA Delete.

Themen

Löschen eines einzelnen Objekts

Sie können die Amazon-S3-Konsole oder die DELETE API verwenden, um ein einzelnes vorhandenes Objekt aus einem S3-Bucket zu löschen. Weitere Informationen zum Löschen von Objekten in Amazon S3 finden Sie unter Löschen von Amazon-S3-Objekten.

Für alle Objekte, die Sie in Ihrem S3-Bucket aufbewahren, entstehen Speicherkosten, deshalb sollten Sie Objekte löschen, die Sie nicht mehr brauchen. Wenn Sie beispielsweise Protokolldateien sammeln, sollten Sie sie unbedingt löschen, wenn sie nicht mehr benötigt werden. Sie können einen Lebenszyklusregel definieren, um Objekte wie Protokolldateien automatisch zu löschen. Weitere Informationen finden Sie unter the section called “Einrichten der Lebenszyklus-Konfiguration”.

Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon-S3-Preise.

Verwenden der S3-Konsole

Befolgen Sie diese Schritte, um mithilfe der Amazon-S3-Konsole ein einzelnes Objekt aus einem Bucket zu löschen.

> **Warning**
>
> Wenn Sie ein Objekt oder eine angegebene Objektversion dauerhaft in der Amazon S3-Konsole löschen, kann das Löschen nicht rückgängig gemacht werden.

So löschen Sie ein Objekt mit aktiviertem oder ausgesetztem Versioning

> **Note**
>
> Wenn die Versions-ID für ein Objekt in einem Bucket mit ausgesetztem Versioning als markiert ist`NULL`, löscht S3 das Objekt dauerhaft, da keine früheren Versionen vorhanden sind. Wenn jedoch eine gültige Versions-ID für das Objekt in einem Bucket mit ausgesetztem Versioning aufgeführt ist, erstellt S3 eine Löschmarkierung für das gelöschte Objekt, wobei die vorherigen Versionen des Objekts beibehalten werden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket Name (Bucket-Name) den Namen des Buckets aus, aus dem Sie ein Objekt löschen möchten.
3. Wählen Sie das Objekt und dann Löschen aus.
4. Um das Löschen der Objektliste unter Angegebene Objekte im Textfeld Objekte löschen? zu bestätigen, geben Sie ein**`delete`**.

So löschen Sie eine bestimmte Objektversion in einem Bucket mit aktiviertem Versioning

> **Warning**
>
> Wenn Sie eine bestimmte Objektversion in Amazon S3 dauerhaft löschen, kann das Löschen nicht rückgängig gemacht werden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket Name (Bucket-Name) den Namen des Buckets aus, aus dem Sie ein Objekt löschen möchten.
3. Wählen Sie das zu löschende Objekt aus.
4. Wählen Sie den Schalter Versionen anzeigen aus.
5. Wählen Sie die Objektversion und dann Löschen aus.
6. Um das dauerhafte Löschen der spezifischen Objektversionen zu bestätigen, die unter Angegebene Objekte aufgeführt sind, geben Sie im Textfeld Objekte löschen? die Option Dauerhaftes Löschen ein. Amazon S3 löscht die spezifische Objektversion dauerhaft.

So löschen Sie dauerhaft ein Objekt in einem Amazon S3-Bucket, für den das Versioning nicht aktiviert ist

> **Warning**
>
> Wenn Sie ein Objekt in Amazon S3 dauerhaft löschen, kann das Löschen nicht rückgängig gemacht werden. Außerdem sind Löschvorgänge für alle Buckets ohne aktiviertes Versioning dauerhaft.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket Name (Bucket-Name) den Namen des Buckets aus, aus dem Sie ein Objekt löschen möchten.
3. Wählen Sie das Objekt und dann Löschen aus.
4. Um das dauerhafte Löschen des unter Angegebene Objekte aufgelisteten Objekts zu bestätigen, geben Sie im Textfeld Objekte löschen? dauerhaft löschen ein.

> **Note**
>
> Wenn Sie Probleme beim Löschen Ihres Objekts haben, finden Sie weitere Informationen unter Ich möchte versionierte Objekte dauerhaft löschen.

Verwenden der AWS SDKs

Die folgenden Beispiele zeigen, wie Sie mit den - AWS SDKs ein Objekt aus einem Bucket löschen können. Weitere Informationen finden Sie unter DELETE Object in der API-Referenz zum Amazon Simple Storage Service.

Wenn für den Bucket das S3-Versioning aktiviert ist, stehen Ihnen die folgenden Optionen zur Verfügung:

- Löschen einer spezifischen Objektversion durch Angabe einer Versions-ID.
- Löschen eines Objekts ohne Angabe einer Version-ID. In diesem Fall fügt Amazon S3 dem Objekt eine Löschmarkierung hinzu.

Weitere Informationen über das S3-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

Java

Example Beispiel 1: Löschen eines Objekts (nicht versionierter Bucket)

Das folgende Beispiel geht davon aus, dass der Bucket nicht versioning-fähig ist und dass das Objekt über keine Versions-IDs verfügt. In der Löschanfrage geben Sie nur den Objektschlüssel und keine Versions-ID an.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectRequest;

import java.io.IOException;

public class DeleteObjectNonVersionedBucket {

    public static void main(String[] args) throws IOException {
```

```
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ****";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(bucketName, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Beispiel 2: Löschen eines Objekts (versionierter Bucket)

Das folgende Beispiel löscht ein Objekt aus einem versionierten Bucket. Das Beispiel löscht eine spezifische Objektversion durch Angabe des Namens des Objektschlüssels und der Versions-ID.

Das Beispiel erledigt Folgendes:

1. Fügt dem Bucket ein Beispielobjekt hinzu. Amazon S3 gibt die die Versions-ID des neu hinzugefügten Objekts zurück. Das Beispiel verwendet diese Versions-ID in der Löschanfrage.
2. Löscht die Objektversion durch Angabe sowohl des Objektschlüsselnamens als auch einer Versions-ID. Wenn keine anderen Versionen dieses Objekts vorhanden sind, löscht Amazon S3 das gesamte Objekt. Andernfalls löscht Amazon S3 die angegebene Version.

> **Note**
>
> Sie können Version-IDs eines Objekts durch Senden einer `ListVersions`-Anfrage anfordern.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.DeleteVersionRequest;
import com.amazonaws.services.s3.model.PutObjectResult;

import java.io.IOException;

public class DeleteObjectVersionEnabledBucket {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ****";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Check to ensure that the bucket is versioning-enabled.
            String bucketVersionStatus =
 s3Client.getBucketVersioningConfiguration(bucketName).getStatus();
            if (!bucketVersionStatus.equals(BucketVersioningConfiguration.ENABLED))
 {
                System.out.printf("Bucket %s is not versioning-enabled.",
 bucketName);
            } else {
                // Add an object.
                PutObjectResult putResult = s3Client.putObject(bucketName, keyName,
                        "Sample content for deletion example.");
                System.out.printf("Object %s added to bucket %s\n", keyName,
 bucketName);

                // Delete the version of the object that we just created.
                System.out.println("Deleting versioned object " + keyName);
```

```
                s3Client.deleteVersion(new DeleteVersionRequest(bucketName, keyName,
 putResult.getVersionId()));
                System.out.printf("Object %s, version %s deleted\n", keyName,
 putResult.getVersionId());
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Das folgende Beispiel veranschaulicht, wie Sie ein Objekt sowohl aus versionierten als auch aus nicht-versionierten Buckets löschen. Weitere Informationen über das S3-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

Example Löschen eines Objekts aus einem nicht-versionierten Bucket

Das folgende C#-Beispiel löscht ein Objekt aus einem nicht-versionierten Bucket. Das Beispiel geht davon aus, dass die Objekte über keine Versions-IDs verfügen. Sie geben daher keine Versions-IDs an. Sie geben nur den Objektschlüssel ein.

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteObjectNonVersionedBucketTest
    {
```

```
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** object key ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            DeleteObjectNonVersionedBucketAsync().Wait();
        }
        private static async Task DeleteObjectNonVersionedBucketAsync()
        {
            try
            {
                var deleteObjectRequest = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };

                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(deleteObjectRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 deleting an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 deleting an object", e.Message);
            }
        }
    }
}
```

Example Löschen eines Objekts aus einem versionierten Bucket

Das folgende C#-Beispiel löscht ein Objekt aus einem versionierten Bucket. Es löscht eine bestimmte Objektversion durch Angabe des Namens des Objektschlüssels und der Versions-ID.

Der Code führt die folgenden Aufgaben durch:

1. Aktiviert das S3-Versioning für einen Bucket, den Sie angeben (wenn S3-Versioning bereits aktiviert ist, hat dies keine Auswirkungen).
2. Fügt dem Bucket ein Beispielobjekt hinzu. In der Antwort gibt Amazon S3 die Versions-ID des neu hinzugefügten Objekts zurück. Das Beispiel verwendet diese Versions-ID in der Löschanfrage.
3. Löscht das Beispielobjekt durch Angabe sowohl des Objektschlüsselnamens als auch einer Versions-ID.

> **Note**
>
> Sie können die Version-IDs eines Objekts durch Senden einer `ListVersions`-Anfrage anfordern.
>
> ```
> var listResponse = client.ListVersions(new ListVersionsRequest { BucketName
>  = bucketName, Prefix = keyName });
> ```

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteObjectVersion
    {
        private const string bucketName = "*** versioning-enabled bucket name ***";
        private const string keyName = "*** Object Key Name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
```

```
        {
            client = new AmazonS3Client(bucketRegion);
            CreateAndDeleteObjectVersionAsync().Wait();
        }

        private static async Task CreateAndDeleteObjectVersionAsync()
        {
            try
            {
                // Add a sample object.
                string versionID = await PutAnObject(keyName);

                // Delete the object by specifying an object key and a version ID.
                DeleteObjectRequest request = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    VersionId = versionID
                };
                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(request);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
deleting an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
deleting an object", e.Message);
            }
        }

        static async Task<string> PutAnObject(string objectKey)
        {
            PutObjectRequest request = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = objectKey,
                ContentBody = "This is the content body!"
            };
            PutObjectResponse response = await client.PutObjectAsync(request);
            return response.VersionId;
```

```
        }
    }
}
```

PHP

Dieses Beispiel zeigt, wie Sie Klassen aus Version 3 des verwenden, AWS SDK for PHP um ein Objekt aus einem nicht versionierten Bucket zu löschen. Weitere Informationen über das Löschen eines Objekts aus einem versionsfähigen Bucket finden Sie unter Verwenden der REST-API.

Dieses Beispiel setzt voraus, dass Sie die Anweisungen für Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen befolgen und das AWS SDK for PHP ordnungsgemäß installiert ist. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen.

Das folgende PHP-Beispiel löscht ein Objekt aus einem Bucket. Da dieses Beispiel zeigt, wie Objekte aus nicht versionsgesteuerten Buckets gelöscht werden, gibt es in der Löschanfrage nur den Bucket-Namen und Objektschlüssel (keine Versions-ID) an.

```
<?php

require 'vendor/autoload.php';

use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// 1. Delete the object from the bucket.
try
{
    echo 'Attempting to delete ' . $keyname . '...' . PHP_EOL;

    $result = $s3->deleteObject([
        'Bucket' => $bucket,
```

```
        'Key'    => $keyname
    ]);

    if ($result['DeleteMarker'])
    {
        echo $keyname . ' was deleted or does not exist.' . PHP_EOL;
    } else {
        exit('Error: ' . $keyname . ' was not deleted.' . PHP_EOL);
    }
}
catch (S3Exception $e) {
    exit('Error: ' . $e->getAwsErrorMessage() . PHP_EOL);
}

// 2. Check to see if the object was deleted.
try
{
    echo 'Checking to see if ' . $keyname . ' still exists...' . PHP_EOL;

    $result = $s3->getObject([
        'Bucket' => $bucket,
        'Key'    => $keyname
    ]);

    echo 'Error: ' . $keyname . ' still exists.';
}
catch (S3Exception $e) {
    exit($e->getAwsErrorMessage());
}
```

Javascript

Dieses Beispiel zeigt, wie Sie Version 3 des verwenden, AWS SDK for JavaScript um ein Objekt zu löschen. Weitere Informationen zu AWS SDK for JavaScript finden Sie unter Verwendung der AWS SDK for JavaScript.

```
import { DeleteObjectCommand } from "@aws-sdk/client-s3";
import { s3Client } from "./libs/s3Client.js" // Helper function that creates Amazon
 S3 service client module.

export const bucketParams = { Bucket: "BUCKET_NAME", Key: "KEY" };

export const run = async () => {
```

```
  try {
    const data = await s3Client.send(new DeleteObjectCommand(bucketParams));
    console.log("Success. Object deleted.", data);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

Verwenden der AWS CLI

Um ein Objekt pro Anfrage zu löschen, verwenden Sie die API `DELETE`. Weitere Informationen hierzu finden Sie unter DELETE Object. Weitere Hinweise zur Verwendung der CLI zum Löschen eines Objekts finden Sie unter delete-object.

Verwenden der REST-API

Sie können die - AWS SDKs verwenden, um ein Objekt zu löschen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Weitere Informationen finden Sie unter DELETE Object in der API-Referenz zum Amazon Simple Storage Service.

Löschen von mehreren Objekten

Für alle Objekte, die Sie in Ihrem S3-Bucket aufbewahren, entstehen Speicherkosten, deshalb sollten Sie Objekte löschen, die Sie nicht mehr brauchen. Wenn Sie beispielsweise Protokolldateien sammeln, sollten Sie sie unbedingt löschen, wenn sie nicht mehr benötigt werden. Sie können einen Lebenszyklusregel definieren, um Objekte wie Protokolldateien automatisch zu löschen. Weitere Informationen finden Sie unter the section called “Einrichten der Lebenszyklus-Konfiguration”.

Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon-S3-Preise.

Sie können die Amazon S3-Konsole, AWS SDKs oder die REST-API verwenden, um mehrere Objekte gleichzeitig aus einem S3-Bucket zu löschen.

Verwenden der S3-Konsole

Befolgen Sie diese Schritte, um mithilfe der Amazon-S3-Konsole mehrere Objekte aus einem Bucket zu löschen.

Warning

- Das Löschen eines angegebenen Objekts kann nicht rückgängig gemacht werden.
- Diese Aktion löscht alle angegebenen Objekte. Warten Sie beim Löschen von Ordnern, bis die Löschaktion abgeschlossen ist, bevor Sie dem Ordner neue Objekte hinzufügen. Andernfalls könnten auch neue Objekte gelöscht werden.
- Beim Löschen von Objekten in einem Bucket ohne aktiviertes Versioning löscht Amazon S3 die Objekte dauerhaft.
- Beim Löschen von Objekten in einem Bucket mit aktiviertem oder ausgesetztem Bucket-Versioning erstellt Amazon S3 Löschmarkierungen. Amazon S3 Weitere Informationen finden Sie unter Arbeiten mit Löschmarkierungen.

So löschen Sie Objekte, für die das Versioning aktiviert oder ausgesetzt ist

Note

Wenn die Versions-IDs für das Objekt in einem Bucket mit ausgesetztem Versioning als markiert sind`NULL`, löscht S3 die Objekte dauerhaft, da keine früheren Versionen vorhanden sind. Wenn jedoch eine gültige Versions-ID für die Objekte in einem Bucket mit ausgesetztem Versioning aufgeführt ist, erstellt S3 Löschmarkierungen für die gelöschten Objekte, wobei die vorherigen Versionen der Objekte beibehalten werden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket-Name den Namen des Buckets aus, aus dem Sie die Objekte löschen möchten.
3. Wählen Sie die Objekte und dann Löschen aus.
4. Um das Löschen der Objektliste unter Angegebene Objekte im Textfeld Objekte löschen? zu bestätigen, geben Sie ein**`delete`**.

So löschen Sie bestimmte Objektversionen dauerhaft in einem Bucket mit aktiviertem Versioning

> **Warning**
>
> Wenn Sie bestimmte Objektversionen in Amazon S3 dauerhaft löschen, kann das Löschen nicht rückgängig gemacht werden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket-Name den Namen des Buckets aus, aus dem Sie die Objekte löschen möchten.
3. Wählen Sie das zu löschende -Objekt aus.
4. Wählen Sie den Schalter Versionen anzeigen aus.
5. Wählen Sie die Objektversionen und dann Löschen aus.
6. Um das dauerhafte Löschen der spezifischen Objektversionen zu bestätigen, die unter Angegebene Objekte aufgeführt sind, geben Sie im Textfeld Objekte löschen? die Option Dauerhaftes Löschen ein. Amazon S3 löscht die spezifischen Objektversionen dauerhaft.

So löschen Sie die Objekte in einem Amazon S3-Bucket, für die das Versioning nicht aktiviert ist, dauerhaft

> **Warning**
>
> Wenn Sie ein Objekt in Amazon S3 dauerhaft löschen, kann das Löschen nicht rückgängig gemacht werden. Außerdem sind Löschvorgänge für alle Buckets ohne aktiviertes Versioning dauerhaft.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket-Name den Namen des Buckets aus, aus dem Sie die Objekte löschen möchten.
3. Wählen Sie die Objekte und dann Löschen aus.
4. Um das dauerhafte Löschen der unter Angegebene Objekte aufgelisteten Objekte zu bestätigen, geben Sie im Textfeld Objekte löschen? dauerhaft löschen ein.

> **Note**
>
> Wenn Sie Probleme beim Löschen Ihrer Objekte haben, finden Sie weitere Informationen unter Ich möchte versionierte Objekte dauerhaft löschen.

Verwenden der AWS SDKs

Beispiele für das Löschen mehrerer Objekte mit den - AWS SDKs finden Sie unter Löschen mehrerer Objekte aus einem Amazon S3-Bucket mithilfe eines - AWS SDK.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Verwenden der REST-API

Sie können die - AWS SDKs verwenden, um mehrere Objekte mithilfe der Multi-Object Delete API zu löschen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden.

Weitere Informationen finden Sie unter Löschen mehrerer Objekte in der API-Referenz zum Amazon Simple Storage Service.

Organisieren, Auflisten und Arbeiten mit Ihren Objekten

In Amazon S3 können Sie Präfixe verwenden, um Ihren Speicher zu organisieren. Ein Präfix ist eine logische Gruppierung der Objekte in einem Bucket. Der Präfixwert ist mit einem Verzeichnisnamen vergleichbar, der Ihnen ermöglicht, ähnliche Daten in demselben Verzeichnis eines Buckets zu speichern. Wenn Sie Objekte programmgesteuert hochladen, können Sie Präfixe verwenden, um Ihre Daten zu organisieren.

In der Amazon-S3-Konsole werden Präfixe als Ordner bezeichnet. Sie können alle Ihre Objekte und Ordner in der S3-Konsole anzeigen, indem Sie zu einem Bucket navigieren. Sie können auch Informationen zu jedem Objekt anzeigen, einschließlich Objekteigenschaften.

Weitere Informationen zum Auflisten und Organisieren Ihrer Daten in Amazon S3 finden Sie in den folgenden Themen.

Themen

Organisieren von Objekten mit Präfixen

Sie können Präfixe verwenden, um die Daten zu organisieren, die Sie in Amazon-S3-Buckets speichern. Ein Präfix ist eine Zeichenfolge am Anfang des Objektschlüsselnamens. Ein Präfix kann eine beliebige Länge haben, abhängig von der maximalen Länge des Objektschlüsselnamens (1.024 Byte). Sie können sich Präfixe als eine Möglichkeit vorstellen, Ihre Daten ähnlich wie Verzeichnisse zu organisieren. Präfixe sind jedoch keine Verzeichnisse.

Die Suche nach Präfix begrenzt die Ergebnisse auf die Schlüssel, die mit dem angegebenen Präfix beginnen. Das Trennzeichen veranlasst Listenoperationen, alle Schlüssel mit gemeinsamem Präfix in einer Zusammenfassungsliste als Ergebnis bereitzustellen.

Zweck der Präfix- und Trennzeichen-Parameter ist es, Ihnen dabei zu helfen, Ihre Schlüssel hierarchisch zu organisieren und dann zu durchsuchen. Dazu wählen Sie zuerst ein Trennzeichen für Ihren Bucket aus, wie beispielsweise einen Schrägstrich (/), das in Ihren Schlüsselnamen voraussichtlich nicht vorkommt. Sie können ein anderes Zeichen als Trennzeichen verwenden. Der Schrägstrich (/) hat keine besonderen Eigenschaften, wird aber sehr häufig als Präfix-Trennzeichen verwendet. Anschließend konstruieren Sie Ihre Schlüsselnamen, indem Sie alle Ebenen der Hierarchie verknüpfen und jede Ebene mit dem Trennzeichen abtrennen.

Wenn Sie beispielsweise Informationen über Städte speichern, könnten Sie diese natürlich nach dem Kontinent, dann nach dem Land und dann nach der Provinz oder dem Staat anordnen. Diese Namen enthalten normalerweise keine Interpunktionszeichen, deshalb könnten Sie den Schrägstrich (/) als Trennzeichen verwenden. Die folgenden Beispiele verwenden einen Schrägstrich (/) als Trennzeichen.

- Europa/Frankreich/Nouvelle-Aquitaine/Bordeaux
- North America/Canada/Quebec/Montreal
- North America/USA/Washington/Bellevue
- North America/USA/Washington/Seattle

Wenn Sie auf diese Weise Daten für jede Stadt in der Welt gespeichert haben, ist es relativ mühsam, einen flachen Schlüssel-Namespace zu verwalten. Durch Verwendung von `Prefix` und `Delimiter` für die Listenvorgänge können Sie die Hierarchie verwenden, die Sie zum Auflisten Ihrer Daten erstellt haben. Um beispielsweise alle Staaten der USA aufzulisten, legen Sie `Delimiter='/'` und `Prefix='North America/USA/'` fest. Um alle Provinzen in Kanada aufzulisten, für die Sie Daten haben, legen Sie `Delimiter='/'` und `Prefix='North America/Canada/'` fest.

Weitere Informationen zu Trennzeichen, Präfixen und verschachtelten Ordnern finden Sie unter Unterschied zwischen Präfixen und verschachtelten Ordnern.

Auflisten von Objekten mit Präfixen und Trennzeichen

Wenn Sie eine Listenanforderung mit einem Trennzeichen ausgeben, können Sie Ihre Hierarchie nur auf einer Ebene durchsuchen und die (möglicherweise Millionen von) Schlüsseln, die auf tieferen Ebenen verschachtelt sind, überspringen und zusammenfassen. Nehmen wir zum Beispiel an, dass Sie einen Bereich (*DOC-EXAMPLE-BUCKET*) mit den folgenden Schlüsseln haben:

`sample.jpg`

`photos/2006/January/sample.jpg`

`photos/2006/February/sample2.jpg`

`photos/2006/February/sample3.jpg`

`photos/2006/February/sample4.jpg`

Der Beispiel-Bucket hat nur das `sample.jpg`-Objekt auf der Root-Ebene. Um nur die Objekte auf Root-Ebene im Bucket aufzulisten, senden Sie eine GET-Anforderung mit dem Trennzeichen (`/`) an den Bucket. In der Antwort gibt Amazon S3 den `sample.jpg`-Objektschlüssel zurück, weil er das Trennzeichen `/` nicht enthält. Alle anderen Schlüssel enthalten das Trennzeichen. Amazon S3 gruppiert diese Schlüssel und gibt ein einziges `CommonPrefixes`-Element mit dem Präfix-Wert `photos/` zurück, eine Unterzeichenfolge vom Anfang dieser Schlüssel bis zum Auftreten des angegebenen Trennzeichens.

Example

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>DOC-EXAMPLE-BUCKET</Name>
  <Prefix></Prefix>
```

```
  <Marker></Marker>
  <MaxKeys>1000</MaxKeys>
  <Delimiter>/</Delimiter>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>sample.jpg</Key>
    <LastModified>2011-07-24T19:39:30.000Z</LastModified>
    <ETag>&quot;d1a7fb5eab1c16cb4f7cf341cf188c3d&quot;</ETag>
    <Size>6</Size>
    <Owner>
      <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
      <DisplayName>displayname</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <CommonPrefixes>
    <Prefix>photos/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

Weitere Hinweise zum programmgesteuerten Auflisten von Objektschlüsseln finden Sie unter Programmgesteuertes Auflisten von Objektschlüsseln.

Programmgesteuertes Auflisten von Objektschlüsseln

In Amazon S3 können Schlüssel nach Präfix aufgelistet werden. Sie können ein allgemeines Präfix für die Namen verwandter Schlüssel wählen und diese Schlüssel mit einem Sonderzeichen markieren, das die Hierarchie begrenzt. Sie können dann die Listenoperation verwenden, um Schlüssel hierarchisch auszuwählen und zu durchsuchen. In ähnlicher Weise werden Dateien in einem Dateisystem in Ordnern gespeichert.

Amazon S3 stellt eine Listenoperation bereit, die Ihnen ermöglicht, die in einem Bucket enthaltenen Schlüssel aufzulisten. Schlüssel werden nach Bucket und Präfix für die Auflistung ausgewählt. Angenommen, wir haben einen Bucket namens "`dictionary`", der einen Schlüssel für jedes englische Wort enthält. Sie können einen Aufruf ausführen, um alle Schlüssel in dem Bucket aufzulisten, die mit dem Buchstaben "q" beginnen. Listenergebnisse werden immer in binärer UTF-8-Reihenfolge zurückgegeben.

Sowohl SOAP- als auch REST-Listenvorgänge geben ein XML-Dokument zurück, das die Namen übereinstimmender Schlüssel sowie Informationen über das von jedem Schlüssel identifizierte Objekt enthält.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Anstatt SOAP zu verwenden, empfehlen wir, entweder die REST-API oder die AWS SDKs zu verwenden.

Gruppen von Schlüsseln, die ein gemeinsames Präfix haben, abgeschlossen mit einem speziellen Trennzeichen, können in einer Auflistung nach diesem gemeinsamen Präfix ausgegeben werden. Auf diese Weise können Anwendungen ihre Schlüssel hierarchisch organisieren und durchsuchen, ähnlich wie bei der Organisation Ihrer Dateien in Verzeichnissen in einem Dateisystem.

Um beispielsweise den dictionary-Bucket zu erweitern, um mehr als nur englische Wörter aufzunehmen, könnten Sie Schlüssel bilden, indem Sie jedem Wort ein Präfix für seine Sprache und ein Trennzeichen voranstellen, wie beispielsweise "`French/logical`". Mit Hilfe dieses Namensschemas und der Funktion zur hierarchischen Auflistung könnten Sie eine Liste nur mit französischen Wörtern abrufen. Sie könnten auch auf oberster Ebene eine Liste verfügbarer Sprachen durchsuchen, ohne alle lexikographischen Schlüssel durchlaufen zu müssen. Weitere Informationen über diesen Aspekt der Auflistung finden Sie unter Organisieren von Objekten mit Präfixen.

REST-API

Falls in Ihrer Anwendung erforderlich, können Sie REST-Anforderungen auch direkt senden. Sie können eine GET-Anforderung senden, um alle oder einen Teil der Objekte in einem Bucket zurückzugeben, oder Auswahlkriterien anwenden, um eine Untermenge der Objekte in einem Bucket zurückzugeben. Weitere Informationen finden Sie unter GET-Bucket(List Objects)-Version-2 in Amazon Simple Storage Service – API-Referenz.

Effiziente Implementierung von Listen

Die Listenleistung wird von der Gesamtzahl der Schlüssel in Ihrem Bucket nicht wesentlich beeinflusst. Es ist auch nicht von der Anwesenheit oder Abwesenheit der Argumente `prefix`, `marker`, `maxkeys` oder `delimiter` betroffen.

Durchlaufen mehrseitiger Ergebnisse

Buckets können nahezu unbegrenzt viele Schlüssel enthalten, die vollständigen Ergebnisse einer Listenabfrage können deshalb extrem umfangreich sein. Um große Ergebnismengen zu verwalten,

unterstützt die Amazon-S3-API eine Paginierung, um sie in mehrere Antworten aufzuteilen. Jede Antwort für Listenschlüssel gibt eine Seite mit bis zu 1000 Schlüsseln zurück, mit einer Angabe, ob die Antwort gekürzt wurde. Sie senden eine Reihe von Anforderungen zum Auflisten von Schlüsseln, bis Sie alle Schlüssel erhalten haben. AWS SDK-Wrapper-Bibliotheken bieten dieselbe Paginierung.

Beispiele

In den folgenden Codebeispielen wird veranschaulicht, wie Sie Objekte in einem S3 Bucket auflisten.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Shows how to list the objects in an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket for which to list
        /// the contents.</param>
        /// <returns>A boolean value indicating the success or failure of the
        /// copy operation.</returns>
        public static async Task<bool> ListBucketContentsAsync(IAmazonS3 client,
string bucketName)
        {
            try
            {
                var request = new ListObjectsV2Request
                {
                    BucketName = bucketName,
                    MaxKeys = 5,
                };

                Console.WriteLine("--------------------------------------");
                Console.WriteLine($"Listing the contents of {bucketName}:");
                Console.WriteLine("--------------------------------------");
```

```
                ListObjectsV2Response response;

                do
                {
                    response = await client.ListObjectsV2Async(request);

                    response.S3Objects
                        .ForEach(obj => Console.WriteLine($"{obj.Key,-35}
{obj.LastModified.ToShortDateString(),10}{obj.Size,10}"));

                    // If the response is truncated, set the request
 ContinuationToken
                    // from the NextContinuationToken property of the response.
                    request.ContinuationToken = response.NextContinuationToken;
                }
                while (response.IsTruncated);

                return true;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error encountered on server.
 Message:'{ex.Message}' getting list of objects.");
                return false;
            }
        }
```

Listen Sie Objekte mit einem Paginator auf.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// The following example lists objects in an Amazon Simple Storage
    /// Service (Amazon S3) bucket.
    /// </summary>
    public class ListObjectsPaginator
    {
```

```
        private const string BucketName = "doc-example-bucket";

        public static async Task Main()
        {
            IAmazonS3 s3Client = new AmazonS3Client();

            Console.WriteLine($"Listing the objects contained in {BucketName}:
\n");
            await ListingObjectsAsync(s3Client, BucketName);
        }

        /// <summary>
        /// This method uses a paginator to retrieve the list of objects in an
        /// an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the S3 bucket whose objects
        /// you want to list.</param>
        public static async Task ListingObjectsAsync(IAmazonS3 client, string
 bucketName)
        {
            var listObjectsV2Paginator = client.Paginators.ListObjectsV2(new
 ListObjectsV2Request
            {
                BucketName = bucketName,
            });

            await foreach (var response in listObjectsV2Paginator.Responses)
            {
                Console.WriteLine($"HttpStatusCode: {response.HttpStatusCode}");
                Console.WriteLine($"Number of Keys: {response.KeyCount}");
                foreach (var entry in response.S3Objects)
                {
                    Console.WriteLine($"Key = {entry.Key} Size = {entry.Size}");
                }
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#       $1 - The name of the bucket.
#
# Returns:
#       The list of files in text format.
#     And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function list_items_in_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api list-objects \
    --bucket "$bucket_name" \
    --output text \
```

```
    --query 'Contents[].{Key: Key, Size: Size}')

  # shellcheck disable=SC2181
  if [[ ${?} -eq 0 ]]; then
    echo "$response"
  else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
  fi
}
```

- API-Details finden Sie unter ListObjectsV2 in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::ListObjects(const Aws::String &bucketName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::ListObjectsRequest request;
    request.WithBucket(bucketName);

    auto outcome = s3_client.ListObjects(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: ListObjects: " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        Aws::Vector<Aws::S3::Model::Object> objects =
                outcome.GetResult().GetContents();
```

```
        for (Aws::S3::Model::Object &object: objects) {
            std::cout << object.GetKey() << std::endl;
        }
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Im folgenden Beispiel wird der `list-objects` Befehl verwendet, um die Namen aller Objekte im angegebenen Bucket anzuzeigen:

```
aws s3api list-objects --bucket text-content --query 'Contents[].{Key: Key, Size:
 Size}'
```

Das Beispiel verwendet das Argument `--query` , um die Ausgabe von auf den Schlüsselwert und die Größe für jedes Objekt `list-objects` zu filtern

Weitere Informationen zu Objekten finden Sie unter Arbeiten mit Amazon S3-Objekten im Amazon S3-Entwicklerhandbuch.

- API-Details finden Sie unter ListObjectsV2 in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// ListObjects lists the objects in a bucket.
func (basics BucketBasics) ListObjects(bucketName string) ([]types.Object, error)
 {
 result, err := basics.S3Client.ListObjectsV2(context.TODO(),
 &s3.ListObjectsV2Input{
  Bucket: aws.String(bucketName),
 })
 var contents []types.Object
 if err != nil {
  log.Printf("Couldn't list objects in bucket %v. Here's why: %v\n", bucketName,
 err)
 } else {
  contents = result.Contents
 }
 return contents, err
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for Go - Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.S3Object;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class ListObjects {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket from which objects are
 read.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBucketObjects(s3, bucketName);
        s3.close();
    }

    public static void listBucketObjects(S3Client s3, String bucketName) {
        try {
            ListObjectsRequest listObjects = ListObjectsRequest
                    .builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.print("\n The name of the key is " + myValue.key());
                System.out.print("\n The object is " + calKb(myValue.size()) + "
 KBs");
                System.out.print("\n The owner is " + myValue.owner());
            }

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // convert bytes to kbs.
    private static long calKb(Long val) {
        return val / 1024;
    }
}
```

Listet Objekte mithilfe der Paginierung auf.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.paginators.ListObjectsV2Iterable;

public class ListObjectsPaginated {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket from which objects are
 read.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBucketObjects(s3, bucketName);
        s3.close();
    }

    public static void listBucketObjects(S3Client s3, String bucketName) {
        try {
            ListObjectsV2Request listReq = ListObjectsV2Request.builder()
                    .bucket(bucketName)
                    .maxKeys(1)
                    .build();

            ListObjectsV2Iterable listRes = s3.listObjectsV2Paginator(listReq);
            listRes.stream()
                    .flatMap(r -> r.contents().stream())
```

```
                    .forEach(content -> System.out.println(" Key: " +
 content.key() + " size = " + content.size()));

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie alle Objekte in Ihrem Bucket auf. Wenn es mehr als ein Objekt gibt, IsTruncated NextContinuationToken wird verwendet, um über die vollständige Liste zu iterieren.

```
import {
  S3Client,
  // This command supersedes the ListObjectsCommand and is the recommended way to
 list objects.
  ListObjectsV2Command,
} from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new ListObjectsV2Command({
    Bucket: "my-bucket",
    // The default and maximum number of keys returned is 1000. This limits it to
    // one for demonstration purposes.
```

```
    MaxKeys: 1,
  });

  try {
    let isTruncated = true;

    console.log("Your bucket contains the following objects:\n");
    let contents = "";

    while (isTruncated) {
      const { Contents, IsTruncated, NextContinuationToken } =
        await client.send(command);
      const contentsList = Contents.map((c) => ` • ${c.Key}`).join("\n");
      contents += contentsList + "\n";
      isTruncated = IsTruncated;
      command.input.ContinuationToken = NextContinuationToken;
    }
    console.log(contents);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for JavaScript -Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listBucketObjects(bucketName: String) {
    val request = ListObjectsRequest {
        bucket = bucketName
    }
```

```
    S3Client { region = "us-east-1" }.use { s3 ->

        val response = s3.listObjects(request)
        response.contents?.forEach { myObject ->
            println("The name of the key is ${myObject.key}")
            println("The object is ${myObject.size?.let { calKb(it) }} KBs")
            println("The owner is ${myObject.owner}")
        }
    }
}

private fun calKb(intValue: Long): Long {
    return intValue / 1024
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum - SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie Objekte in einem Bucket auf.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $contents = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            echo "The contents of your bucket are: \n";
            foreach ($contents['Contents'] as $content) {
                echo $content['Key'] . "\n";
            }
```

```
        } catch (Exception $exception) {
            echo "Failed to list objects in $this->bucketName with error: " .
$exception->getMessage();
            exit("Please fix error with listing objects before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    @staticmethod
    def list(bucket, prefix=None):
        """
        Lists the objects in a bucket, optionally filtered by a prefix.

        :param bucket: The bucket to query. This is a Boto3 Bucket resource.
        :param prefix: When specified, only objects that start with this prefix
 are listed.
```

```
        :return: The list of objects.
        """
        try:
            if not prefix:
                objects = list(bucket.objects.all())
            else:
                objects = list(bucket.objects.filter(Prefix=prefix))
            logger.info(
                "Got objects %s from bucket '%s'", [o.key for o in objects],
bucket.name
            )
        except ClientError:
            logger.exception("Couldn't get objects for bucket '%s'.",
bucket.name)
            raise
        else:
            return objects
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum - SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketListObjectsWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
  def initialize(bucket)
```

```
    @bucket = bucket
  end

  # Lists object in a bucket.
  #
  # @param max_objects [Integer] The maximum number of objects to list.
  # @return [Integer] The number of objects listed.
  def list_objects(max_objects)
    count = 0
    puts "The objects in #{@bucket.name} are:"
    @bucket.objects.each do |obj|
      puts "\t#{obj.key}"
      count += 1
      break if count == max_objects
    end
    count
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't list objects in bucket #{bucket.name}. Here's why:
 #{e.message}"
    0
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"

  wrapper = BucketListObjectsWrapper.new(Aws::S3::Bucket.new(bucket_name))
  count = wrapper.list_objects(25)
  puts "Listed #{count} objects."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_objects(client: &Client, bucket: &str) -> Result<(), Error> {
    let mut response = client
        .list_objects_v2()
        .bucket(bucket.to_owned())
        .max_keys(10) // In this example, go 10 at a time.
        .into_paginator()
        .send();

    while let Some(result) = response.next().await {
        match result {
            Ok(output) => {
                for object in output.contents() {
                    println!(" - {}", object.key().unwrap_or("Unknown"));
                }
            }
            Err(err) => {
                eprintln!("{err:?}")
            }
        }
    }

    Ok(())
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum - SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    TRY.
        oo_result = lo_s3->listobjectsv2(         " oo_result is returned for
testing purposes. "
          iv_bucket = iv_bucket_name
        ).
        MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum - SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func listBucketFiles(bucket: String) async throws -> [String] {
        let input = ListObjectsV2Input(
            bucket: bucket
        )
        let output = try await client.listObjectsV2(input: input)
        var names: [String] = []

        guard let objList = output.contents else {
            return []
        }

        for obj in objList {
            if let objName = obj.key {
                names.append(objName)
            }
        }

        return names
    }
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum - SDK für Swift.

Organisieren von Objekten in der Amazon S3-Konsole mithilfe von Ordnern

In Amazon S3 sind Buckets und Objekte die primären Ressourcen, in denen Objekte in Buckets gespeichert werden. Amazon S3 besitzt eine flache Struktur statt einer Hierarchie, wie Sie sie in der Regel aus Dateisystemen kennen. Um eine möglichst einfache Organisation zu ermöglichen, unterstützt die Amazon-S3-Konsole jedoch das Ordner-Konzept, um Objekte gruppieren zu können. Die Konsole verwendet dazu ein gemeinsames Namenspräfix für die gruppierten Objekte. Mit anderen Worten, die gruppierten Objekte haben Namen, die mit einer bestimmten Zeichenfolge beginnen. Diese gemeinsame Zeichenfolge oder das gemeinsame Präfix ist der Ordnername. Objektnamen werden auch als Schlüsselnamen bezeichnet.

Beispielsweise können Sie einen Ordner namens `photos` in der Konsole erstellen und ein Objekt namens `myphoto.jpg` darin speichern. Das Objekt wird dann mit dem Schlüsselnamen `photos/myphoto.jpg` gespeichert, wobei `photos/` das Präfix ist.

Nachfolgend zwei weitere Beispiele:

- Wenn Sie drei Objekte in Ihrem Bucket haben – `logs/date1.txt`, `logs/date2.txt` und `logs/date3.txt`–, zeigt die Konsole einen Ordner namens `logs` an. Wenn Sie den Ordner in der Konsole öffnen, sehen Sie drei Objekte: `date1.txt`, `date2.txt` und `date3.txt`.
- Wenn Sie ein Objekt namens `photos/2017/example.jpg` haben, zeigt die Konsole den Ordner `photos` an, der den Ordner `2017` enthält. Der Ordner `2017` enthält das Objekt `example.jpg`.

Sie können Ordner innerhalb von Ordnern anlegen, aber keine Buckets innerhalb von Buckets. Sie können Objekte direkt in einen Ordner hochladen und kopieren. Ordner können erstellt, gelöscht und veröffentlicht werden, sie können jedoch nicht umbenannt werden. Objekte können von einem Ordner in einen anderen kopiert werden.

> **Important**
>
> Wenn Sie einen Ordner in Amazon S3 anlegen, erstellt S3 ein 0-Byte-Objekt mit einem Schlüssel, der auf den von Ihnen angegebenen Ordnernamen festgelegt ist. Wenn Sie beispielsweise einen Ordner mit dem Namen `photos` in Ihrem Bucket erstellen, erstellt die Amazon-S3-Konsole ein 0-Byte-Objekt mit dem Schlüssel `photos/`. Die Konsole erstellt dieses Objekt, um das Ordnerkonzept zu unterstützen.
>
> Die Amazon-S3-Konsole behandelt alle Objekte mit einem Schrägstrich (/) als letztes Zeichen im Schlüsselnamen als Ordner (z. B. `examplekeyname/`). Es ist nicht möglich, ein Objekt über die Amazon-S3-Konsole hochzuladen, dessen Schlüsselname mit einem / endet. Sie können jedoch Objekte mit einem nachfolgenden Namen über die (AWS CLI), AWS SDKs oder die REST-API / mit der AWS Command Line Interface Amazon S3-API hochladen.
>
> Ein Objekt mit einem angehängten / im Namen wird in der Amazon-S3-Konsole als Ordner angezeigt. Die Amazon-S3-Konsole zeigt für diese Objekte keinen Inhalt und keine Metadaten an. Wenn Sie ein Objekt mit einem angehängten / im Namen kopieren, wird am Zielstandort ein neuer Ordner erstellt, aber die Daten und Metadaten des Objekts werden nicht kopiert.

Themen

Erstellen eines Ordners

In diesem Abschnitt wird beschrieben, wie Sie mit der Amazon-S3-Konsole einen Ordner erstellen.

> **Important**
>
> Wenn Ihre Bucket-Richtlinie das Hochladen von Objekten in diesen Bucket ohne Markierungen, Metadaten oder Zugriffssteuerungsliste (ACL)-Berechtigungsempfänger verhindert, können Sie mit der folgenden Vorgehensweise keinen Ordner erstellen. Laden Sie stattdessen einen leeren Ordner hoch und geben Sie die folgenden Einstellungen in der Upload-Konfiguration an.

Einen Ordner erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, in dem Sie einen Ordner erstellen möchten.
4. Wenn Ihre Bucket-Richtlinie das Hochladen von Objekten ohne Verschlüsselung in diesen Bucket verhindert, müssen Sie Enable (Aktivieren) unter Server-side encryption (Serverseitige Verschlüsselung) wählen.
5. Wählen Sie Create folder.
6. Geben Sie einen Namen für den Ordner ein (z. B. **`favorite-pics`**). Wählen Sie dann Create folder (Ordner erstellen).

Veröffentlichen von Ordnern

Wir empfehlen, jeden öffentlichen Zugriff auf Ihre Amazon-S3-Ordner und Buckets zu blockieren, sofern Sie nicht speziell einen öffentlichen Ordner oder Bucket benötigen. Wenn Sie einen Ordner öffentlich machen, kann jeder Benutzer im Internet alle im Ordner gruppierten Objekte Anzeigen.

In der Amazon-S3-Konsole können Sie einen Ordner öffentlich machen. Außerdem können Sie einen Ordner öffentlich machen, indem Sie eine Bucket-Richtlinie erstellen, die den Datenzugriff durch ein Präfix einschränkt. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3.

> **Warning**
>
> Nachdem Sie einen Ordner in der Amazon-S3-Konsole öffentlich gemacht haben, kann er nicht mehr auf privat gesetzt werden. Stattdessen müssen Sie Berechtigungen für jedes einzelne Objekt im öffentlichen Ordner festlegen, damit die Objekte keinen öffentlichen Zugriff haben. Weitere Informationen finden Sie unter Konfigurieren von ACLs.

Themen

Berechnen der Ordnergröße

In diesem Abschnitt wird beschrieben, wie Sie mit der Amazon-S3-Konsole die Größe eines Ordners berechnen.

So berechnen Sie die Größe eines Ordners

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, in dem Ihr Ordner gespeichert ist.
4. Aktivieren Sie in der Liste Objects (Objekte) das Kontrollkästchen neben dem Namen des Ordners.
5. Wählen Sie Actions (Aktionen) und dann Calculate total size (Gesamtgröße berechnen) aus.

> **Note**
>
> Wenn Sie die Seite verlassen, sind die Ordnerinformationen (einschließlich der Gesamtgröße) danach nicht mehr verfügbar. Sie müssen die Gesamtgröße erneut berechnen, wenn Sie sie erneut sehen möchten.

Important

- Wenn Sie die Aktion Calculate total size (Gesamtgröße berechnen) für bestimmte Objekte oder Ordner in Ihrem Bucket verwenden, berechnet Amazon S3 die Gesamtzahl der Objekte und die Gesamtspeichergröße. Unvollständige oder in Bearbeitung befindliche mehrteilige Uploads und vorherige oder nicht aktuelle Versionen werden jedoch nicht auf die Gesamtzahl der Objekte oder die Gesamtgröße angerechnet. Diese Aktion berechnet nur die Gesamtzahl der Objekte und die Gesamtgröße für die aktuelle oder neueste Version jedes Objekts, das im Bucket gespeichert ist.

  Wenn es beispielsweise zwei Versionen eines Objekts in Ihrem Bucket gibt, zählt der Speicherrechner in Amazon S3 diese nur als ein Objekt. Daher kann die Gesamtzahl der Objekte, die in der Amazon S3-Konsole berechnet wird, von der in S3 Storage Lens angezeigten Metrik Object Count und von der von der Amazon-CloudWatch Metrik gemeldeten Anzahl abweichen`NumberOfObjects`. Ebenso kann sich die Gesamtspeichergröße auch von der in S3 Storage Lens angezeigten Gesamtspeichermetrik und von der in angezeigten `BucketSizeBytes` Metrik unterscheiden CloudWatch.
- Wenn die Berechnung der Gesamtgröße eines großen Ordners zu lange dauert, sollten Sie Amazon S3 Inventory und Amazon S3 Select als Alternative in Betracht ziehen. Erstellen Sie zunächst eine S3-Inventory-Konfiguration, um die Größenmetadaten für jedes Objekt des großen Ordners in einen Inventarbericht aufzunehmen. Es könnte bis zu 48 Stunden dauern, bis der erste S3-Inventarbericht bereitgestellt wird. Wenn der Inventarbericht veröffentlicht ist, fragen Sie ihn mit einem S3 Select `SUM`-Ausdruck ab, um die Größen der Objekte im Ordner zu aggregieren. Weitere Informationen finden Sie unter Konfigurieren des Bestands mit der S3-Konsole und SUMBeispiel für .

Löschen von Ordnern

In diesem Abschnitt erfahren Sie, wie Sie mit der Amazon-S3-Konsole Ordner aus einem S3-Bucket löschen.

Informationen zu den Funktionen und Preisen von Amazon S3 finden Sie unter Amazon S3.

Ordner aus einem S3-Bucket löschen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, aus dem Sie Ordner löschen möchten.
3. Aktivieren Sie in der Liste Objects (Objekte) das Kontrollkästchen neben den Ordnern und Objekten, die Sie löschen möchten.
4. Wählen Sie Löschen aus.
5. Stellen Sie auf der Seite Delete objects (Objekte löschen) sicher, dass die Namen der Ordner, die Sie zum Löschen ausgewählt haben, aufgeführt sind.
6. Geben Sie im Feld Delete objects (Objekte löschen) ein **`delete`**, und wählen Sie Delete objects (Objekte löschen) aus.

> **Warning**
>
> Diese Aktion löscht alle angegebenen Objekte. Warten Sie beim Löschen von Ordnern, bis die Löschaktion abgeschlossen ist, bevor Sie dem Ordner neue Objekte hinzufügen. Andernfalls könnten auch neue Objekte gelöscht werden.

Anzeigen einer Objektübersicht in der Amazon-S3-Konsole

Sie können die Amazon-S3-Konsole verwenden, um einen Überblick über ein Objekt zu erhalten. Die Konsole liefert alle wesentlichen Informationen für ein Objekt an einem Ort.

So öffnen Sie die Detailseite für ein Objekt

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
3. Wählen Sie in der Liste Objects (Objekte) den Namen des Objekts aus, für das Sie eine Übersicht erhalten möchten.

   Die Seite mit Objektdetails wird geöffnet.

4. Um das Objekt herunterzuladen, wählen Sie Object actions (Objektaktionen) und dann Download (Herunterladen). Um den Pfad des Objekts in die Zwischenablage zu kopieren, wählen Sie unter Object URL (Objekt-URL) die URL aus.
5. Wenn Ihr Bucket versionsfähig ist, können Sie Versions (Versionen) wählen, um alle Versionen des Objekts anzuzeigen.

   - Um eine Objektversion herunterzuladen, aktivieren Sie das Kontrollkästchen neben der Versions-ID, wählen Sie Actions (Aktionen) und dann Download (Herunterladen).
   - Um eine Objektversion zu löschen, aktivieren Sie das Kontrollkästchen neben der Versions-ID und wählen Sie Delete (Löschen).

> **Important**
>
> Sie können den Löschvorgang für ein Objekt nur rückgängig machen, wenn seine aktuelle Version gelöscht wurde. Es ist nicht möglich, das Löschen einer vorherigen Version eines Objekts rückgängig zu machen, das gelöscht wurde.

Anzeigen von Objekteigenschaften in der Amazon-S3-Konsole

Sie können die Amazon-S3-Konsole verwenden, um die Eigenschaften eines Objekts anzuzeigen, einschließlich Speicherklasse, Verschlüsselungs-Einstellungen, Markierungen und Metadaten.

Die Eigenschaften für ein Objekt anzeigen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
3. Wählen Sie in der Liste Objects (Objekte) den Namen des Objekts aus, dessen Eigenschaften Sie anzeigen wollen.

   Die Object overview (Objektübersicht) für Ihr Objekt wird geöffnet. Sie können nach unten scrollen, um die Objekteigenschaften anzuzeigen.
4. Auf der Seite Object overview (Objektübersicht) können Sie die folgenden Eigenschaften für das Objekt konfigurieren.

> **Note**
>
> Wenn Sie die Eigenschaften Storage Class (Speicherklasse), Encryption (Verschlüsselung) oder Metadata (Metadaten) ändern, wird ein neues Objekt erstellt, um das alte zu ersetzen. Wenn S3-Versioning aktiviert ist, wird eine neue Version des Objekts erstellt, und das vorhandene Objekt wird zu einer älteren Version. Die Rolle, die die Eigenschaft ändert, wird auch Besitzer des neuen Objekts (oder der neuen Objektversion).

a. Storage class (Speicherklasse) – Jedem Objekt in Amazon S3 ist eine Speicherklasse zugeordnet. Welche Speicherklasse Sie wählen, ist davon abhängig, wie häufig Sie auf das Objekt zugreifen. Die Standardspeicherklasse für S3-Objekte ist STANDARD. Sie wählen, welche Speicherlasse verwendet werden soll, wenn Sie ein Objekt hochladen. Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.

   Um die Speicherklasse nach dem Hochladen eines Objekts zu ändern, wählen Sie Storage class (Speicherklasse). Wählen Sie die gewünschte Speicherklasse aus, und wählen Sie dann Save (Speichern).

b. Serverseitige Verschlüsselungseinstellungen — Sie können serverseitige Verschlüsselung verwenden, um Ihre S3-Objekte zu verschlüsseln. Weitere Informationen finden Sie unter Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS) oder Angeben serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

c. Metadata (Metadaten) – Jedes Objekt in Amazon S3 weist eine Gruppe von Namen-Wert-Paaren auf, die seine Metadaten repräsentieren. Weitere Informationen zum Hinzufügen von Metadaten zu einem S3-Objekt finden Sie unter Bearbeiten von Objektmetadaten in der Amazon-S3-Konsole.

d. Tags – Sie kategorisieren den Speicher, indem Sie einem S3-Objekt Tags hinzufügen. Weitere Informationen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

e. Rechtliche Aufbewahrung und Vermietung von Objektsperre – Sie können verhindern, dass ein Objekt gelöscht wird. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

Arbeiten mit vorsignierten URLs

Um zeitlich begrenzten Zugriff auf Objekte in Amazon S3 zu gewähren, ohne Ihre Bucket-Richtlinie zu aktualisieren, können Sie vorsignierte URLs verwenden. Eine vorsignierte URL kann in einem Browser eingegeben oder von einem Programm verwendet werden, um ein Objekt herunterzuladen. Die von der vorsignierten URL verwendeten Anmeldeinformationen sind die des AWS Benutzers, der die URL generiert hat.

Sie können auch vorsignierte URLs verwenden, um es einer Person zu ermöglichen, ein bestimmtes Objekt in Ihren Amazon-S3-Bucket hochzuladen. Dies ermöglicht einen Upload, ohne dass eine andere Partei über AWS Sicherheitsanmeldeinformationen oder Berechtigungen verfügen muss. Wenn im Bucket bereits ein Objekt mit demselben Schlüssel vorhanden ist, wie in der vorsignierten URL angegeben wird, ersetzt Amazon S3 das vorhandene Objekt durch das hochgeladene Objekt.

Sie können die vorsignierte URL mehrmals verwenden, bis hin zum Ablaufdatum und -zeitpunkt.

Wenn Sie eine vorsignierte URL erstellen, müssen Sie Ihre Sicherheitsanmeldedaten eingeben und dann Folgendes angeben:

- Ein Amazon-S3-Bucket
- Ein Objektschlüssel (Herunterladen: dieses Objekt wird sich in Ihrem Amazon-S3-Bucket befinden; Hochladen: der Name der hochzuladenden Datei)
- Eine HTTP-Methode (`GET` zum Herunterladen oder `PUT` zum Hochladen von Objekten)
- Ein Ablaufzeitintervall

Derzeit unterstützen vorsignierte Amazon-S3-URLs die Verwendung der folgenden Prüfsummenalgorithmen für die Datenintegrität (CRC32, CRC32C, SHA-1, SHA-256) nicht, wenn Sie Objekte hochladen. Zur Überprüfung der Integrität Ihres Objekts nach dem Hochladen können Sie einen MD5-Digest des Objekts bereitstellen, wenn Sie es mit einer vorsignierten URL hochladen. Weitere Informationen über die Objektintegrität finden Sie unter Überprüfung der Objektintegrität.

Themen

Wer eine vorsignierte URL erstellen kann

Alle Benutzer mit gültigen Sicherheitsanmeldeinformationen können vorsignierte URLs erstellen. Um erfolgreich auf ein Objekt zugreifen zu können, muss die vorsignierte URL von jemandem erstellt werden, der die Berechtigung für den Vorgang besitzt, auf dem die vorsignierte URL basiert.

Die Arten von Anmeldeinformationen, die Sie zum Erstellen einer vorsignierten URL verwenden können:

- IAM-Instance-Profil – Bis zu 6 Stunden gültig.
- AWS Security Token Service: Bei einer Unterzeichnung mit langfristigen Sicherheitsanmeldedaten bis zu 36 Stunden oder für die Dauer der temporären Anmeldeinformationen gültig, je nachdem, was zuerst endet.
- IAM-Benutzer – Bis zu 7 Tage gültig, wenn Sie AWS Signature Version 4 verwenden.

  Um eine vorsignierte URL zu erstellen, die bis zu 7 Tage gültig ist, delegieren Sie zunächst die IAM-Benutzer-Anmeldeinformationen (den Zugriffsschlüssel und den geheimen Schlüssel) an die Methode, die Sie verwenden, um die vorsignierte URL zu erstellen.

> **Note**
>
> Wenn Sie eine vorsignierte URL mit temporären Anmeldeinformationen erstellt haben, verfällt die URL mit Ablauf der Anmeldeinformationen. Dies gilt auch dann, wenn die URL mit einer späteren Ablaufzeit erstellt wurde. Informationen zu den Lebensdauerzeiten temporärer Sicherheitsanmeldeinformationen finden Sie unter Vergleichen von AWS STS - API-Operationen im IAM-Benutzerhandbuch.

Ablaufzeit für vorsignierte URLs

Eine vorsignierte URL bleibt für den Zeitraum gültig, der bei der Generierung der URL angegeben wurde. Wenn Sie eine vorsignierte URL mit der Amazon-S3-Konsole erstellen, kann die Ablaufzeit auf einen Zeitraum zwischen 1 Minute und 12 Stunden festgelegt werden. Wenn Sie die AWS CLI oder AWS SDKs verwenden, kann die Ablaufzeit auf bis zu 7 Tage festgelegt werden.

Wenn Sie eine vorsignierte URL mit einem temporären Token erstellt haben, läuft die URL ab, wenn das Token abläuft, auch wenn Sie die URL mit einer späteren Ablaufzeit erstellt haben. Weitere

Informationen darüber, wie sich die von Ihnen verwendeten Anmeldeinformationen auf die Ablaufzeit auswirken, finden Sie unter Wer eine vorsignierte URL erstellen kann.

Amazon S3 überprüft das Ablaufdatum und die Ablaufzeit in einer signierten URL zum Zeitpunkt der HTTP-Anforderung. Beginnt ein Client beispielsweise mit dem Herunterladen einer großen Datei unmittelbar vor der Ablaufzeit, wird der Download auch dann fortgesetzt, wenn die Ablaufzeit während des Downloads verstreicht. Wenn die Verbindung jedoch unterbrochen wird und der Client versucht, den Download nach Ablauf der Zeit erneut zu starten, schlägt der Download fehl.

Beschränkung der Funktionen für vorsignierte URLs

Die Funktionen einer vorsignierten URL sind durch die Berechtigungen des Benutzers eingeschränkt, der sie erstellt hat. Im Wesentlichen sind vorsignierte URLs Inhaber-Token, die denjenigen, die sie besitzen, Zugriff gewähren. Daher empfehlen wir Ihnen, sie angemessen zu schützen. Im Folgenden finden Sie einige Methoden, die Sie verwenden können, um die Verwendung Ihrer vorsignierten URLs zu beschränken.

AWS Signature Version 4 (SigV4)

Um ein bestimmtes Verhalten zu erzwingen, wenn vorsignierte URL-Anfragen mit AWS Signature Version 4 (SigV4) authentifiziert werden, können Sie Bedingungsschlüssel in Bucket-Richtlinien und Zugriffspunkt-Richtlinien verwenden. Beispielsweise verwendet die folgende Bucket-Richtlinie die `s3:signatureAge`-Bedingung, um jede vorsignierte URL-Anfrage von Amazon S3 für Objekte im *`DOC-EXAMPLE-BUCKET1`*-Bucket zu verweigern, wenn die Signatur mehr als 10 Minuten alt ist. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Deny a presigned URL request if the signature is more than 10 min
 old",
         "Effect": "Deny",
         "Principal": {"AWS":"*"},
         "Action": "s3:*",
         "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
         "Condition": {
            "NumericGreaterThan": {
               "s3:signatureAge": 600000
            }
```

```
            }
        }
    ]
}
```

Weitere Informationen zu Richtlinienschlüsseln im Zusammenhang mit AWS Signature Version 4 finden Sie unter AWS Authentifizierung mit Signature Version 4 in der API-Referenz zu Amazon Simple Storage Service.

Beschränkung der Netzwege

Wenn Sie die Verwendung vorsignierter URLs und des gesamten Amazon S3-Zugriffs auf bestimmte Netzwerkpfade einschränken möchten, können Sie AWS Identity and Access Management (IAM)-Richtlinien schreiben. Diese Richtlinien können für den IAM-Prinzipal, der den Aufruf vornimmt, den Amazon-S3-Bucket oder beide festgelegt werden.

Eine Netzwerkpfadbeschränkung für den IAM-Prinzipal erfordert, dass der Benutzer dieser Anmeldeinformationen Anfragen aus dem angegebenen Netzwerk stellt. Eine Einschränkung des Buckets oder des Zugriffspunkts erfordert, dass alle Anfragen an diese Ressource aus dem angegebenen Netz stammen. Diese Einschränkungen gelten auch außerhalb des Szenarios der vorsignierten URL.

Der globale IAM-Bedingungsschlüssel, den Sie verwenden, hängt von der Art des Endpunkts ab. Wenn Sie den öffentlichen Endpunkt für Amazon S3 verwenden, verwenden Sie `aws:SourceIp`. Wenn Sie einen Virtual Private Cloud (VPC)-Endpunkt für Amazon S3 nutzen, verwenden Sie `aws:SourceVpc` oder `aws:SourceVpce`.

Die folgende IAM-Richtlinienanweisung erfordert, dass der Prinzipal AWS nur aus dem angegebenen Netzwerkbereich auf zugreift. Mit dieser Richtlinie müssen alle Zugriffe von diesem Bereich ausgehen. Dies gilt auch für den Fall, dass jemand eine vorsignierte URL für Amazon S3 verwendet. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
{
    "Sid": "NetworkRestrictionForIAMPrincipal",
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
        "NotIpAddressIfExists": {"aws:SourceIp": "IP-address-range"},
        "BoolIfExists": {"aws:ViaAWSService": "false"}
```

```
    }
}
```

Weitere Beispiele für Bucket-Richtlinien, die den `aws:SourceIp` AWS globalen Bedingungsschlüssel verwenden, um den Zugriff auf einen Amazon S3-Bucket auf einen bestimmten Netzwerkbereich zu beschränken, finden Sie unter Verwalten des Zugriffs auf der Grundlage bestimmter IP-Adressen.

Gemeinsame Nutzung von Objekten mit vorsignierten URLs

Standardmäßig sind alle Amazon-S3-Objekte privat und nur der Objekteigentümer hat die Berechtigung, darauf zuzugreifen. Der Objekteigentümer kann Objekte jedoch mit anderen teilen, indem er eine vorsignierte URL erstellt. Eine vorsignierte URL verwendet Sicherheitsanmeldeinformationen, um eine zeitlich begrenzte Berechtigung zum Herunterladen von Objekten zu gewähren. Die URL kann in einem Browser eingegeben oder von einem Programm verwendet werden, um das Objekt herunterzuladen. Die von der vorsignierten URL verwendeten Anmeldeinformationen sind die des AWS Benutzers, der die URL generiert hat.

Allgemeine Informationen zu vorsignierten URLs finden Sie unter Arbeiten mit vorsignierten URLs.

Sie können eine vorsignierte URL für die Freigabe eines Objekts erstellen, ohne Code schreiben zu müssen, indem Sie die Amazon S3-Konsole, AWS Explorer für Visual Studio (Windows) oder verwenden AWS Toolkit for Visual Studio Code. Sie können eine vorsignierte URL auch programmgesteuert mithilfe der AWS Command Line Interface (AWS CLI) oder der AWS SDKs generieren.

Verwenden der S3-Konsole

Sie können die Amazon-S3-Konsole verwenden, um eine vorsignierte URL für ein Objekt zu generieren, indem Sie diese Schritte ausführen. Wenn Sie die Konsole nutzen, beträgt die maximale Ablaufzeit für eine vorsignierte URL 12 Stunden ab dem Zeitpunkt ihrer Erstellung.

So generieren Sie eine vorsignierte URL mit der Amazon-S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets mit dem Objekt aus, für das Sie eine vorsignierte URL haben möchten.

4. In der Liste Objekte wählen Sie das Objekt aus, für das Sie eine vorsignierte URL erstellen möchten.
5. Wählen Sie im Menü Objektaktionen die Option An vorsignierte URL freigeben aus.
6. Geben Sie an, wie lange die vorsignierte URL gültig sein soll.
7. Wählen Sie Create presigned URL (Vorsignierte URL erstellen).
8. Wenn eine Bestätigung angezeigt wird, wird die URL automatisch in Ihre Zwischenablage kopiert. Sie sehen eine Schaltfläche zum Kopieren der vorsignierten URL, wenn Sie sie erneut kopieren müssen.

Verwenden der AWS CLI

Der folgende AWS CLI Beispielbefehl generiert eine vorsignierte URL für die Freigabe eines Objekts aus einem Amazon S3-Bucket. Wenn Sie die verwenden AWS CLI, beträgt die maximale Ablaufzeit für eine vorsignierte URL 7 Tage ab dem Zeitpunkt der Erstellung. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3 presign s3://DOC-EXAMPLE-BUCKET1/mydoc.txt --expires-in 604800
```

> **Note**
>
> Für alle , die nach dem 20. März 2019 AWS-Regionen gestartet wurden, müssen Sie die `endpoint-url` und `AWS-Region` mit der -Anforderung angeben. Eine Liste aller Amazon-S3-Regionen und Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS -Referenz.

```
aws s3 presign s3://DOC-EXAMPLE-BUCKET1/mydoc.txt --expires-in 604800 --region af-south-1 --endpoint-url https://s3.af-south-1.amazonaws.com
```

Weitere Informationen finden Sie unter presign in der Referenz zum AWS CLI -Befehl.

Verwenden der AWS SDKs

Beispiele für die Verwendung der - AWS SDKs zum Generieren einer vorsignierten URL für die Freigabe eines Objekts finden Sie unter Erstellen einer vorsignierten URL für Amazon S3 mithilfe eines - AWS SDK.

Wenn Sie die - AWS SDKs verwenden, um eine vorsignierte URL zu generieren, beträgt die maximale Ablaufzeit 7 Tage ab dem Zeitpunkt der Erstellung.

> **Note**
>
> Für alle , die nach dem 20. März 2019 AWS-Regionen gestartet wurden, müssen Sie die `endpoint-url` und `AWS-Region` mit der -Anforderung angeben. Eine Liste aller Amazon-S3-Regionen und Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS -Referenz.

> **Note**
>
> Bei Verwendung der - AWS SDKs muss das Markierungsattribut ein Header und kein Abfrageparameter sein. Alle anderen Attribute können als Parameter für die vorsignierte URL übergeben werden.

Verwenden der AWS Toolkit for Visual Studio (Windows)

> **Note**
>
> Derzeit unterstützt Visual Studio für Mac AWS Toolkit for Visual Studio nicht.

1. Installieren Sie die AWS Toolkit for Visual Studio mit den folgenden Anweisungen: Installieren und Einrichten des Toolkit for Visual Studio im AWS Toolkit for Visual Studio -Benutzerhandbuch.
2. Stellen Sie eine Verbindung zu her, AWS indem Sie die folgenden Schritte ausführen: Herstellen einer Verbindung mit AWS im AWS Toolkit for Visual Studio -Benutzerhandbuch.
3. Doppelklicken Sie im linken Bereich mit der Bezeichnung AWS Explorer auf den Bucket, der Ihr Objekt enthält.
4. Klicken Sie mit der rechten Maustaste auf das Objekt, für das Sie eine vorsignierte URL generieren möchten, und wählen Sie Vorsignierte URL erstellen... aus.
5. Legen Sie im Popup-Fenster das Ablaufdatum und die Ablaufzeit für Ihre vorsignierte URL fest.
6. Der Objektschlüssel , sollte basierend auf dem ausgewählten Objekt vorausgefüllt werden.
7. Wählen Sie GET, um anzugeben, dass diese vorsignierte URL zum Herunterladen eines Objekts verwendet wird.

8. Wählen Sie die Schaltfläche Generate (Generieren) aus.
9. Wählen Sie zum Kopieren der URL in die Zwischenablage Copy (Kopieren) aus.
10. Um die generierte vorsignierte URL zu verwenden, fügen Sie die URL in einen beliebigen Browser ein.

Verwenden von AWS Toolkit for Visual Studio Code

Wenn Sie Visual Studio Code verwenden, können sie AWS Toolkit for Visual Studio Code verwenden, um eine vorsignierte URL zur Freigabe eines Objekts zu erstellen, ohne Code schreiben zu müssen. Allgemeine Informationen finden Sie unter AWS Toolkit for Visual Studio Code im AWS Toolkit for Visual Studio Code -Benutzerhandbuch.

Anweisungen zur Installation der AWS Toolkit for Visual Studio Code finden Sie unter Installieren der AWS Toolkit for Visual Studio Code im AWS Toolkit for Visual Studio Code -Benutzerhandbuch.

1. Stellen Sie eine Verbindung zu her, AWS indem Sie die folgenden Schritte ausführen: Herstellen einer Verbindung mit AWS Toolkit for Visual Studio Code im AWS Toolkit for Visual Studio Code -Benutzerhandbuch.
2. Wählen Sie das AWS Logo im linken Bereich in Visual Studio Code aus.
3. Wählen Sie unter EXPLORER S3 aus.
4. Wählen Sie einen Bucket und eine Datei und öffnen Sie das Kontextmenü (rechte Maustaste).
5. Wählen Sie Vorsignierte URL generieren aus und legen Sie dann die Ablaufzeit (in Minuten) fest.
6. Drücken Sie die Eingabetaste, wodurch die vorsignierte URL in Ihre Zwischenablage kopiert wird.

Hochladen von Objekten mit vorsignierten URLs

Sie können vorsignierte URLs verwenden, damit jemand ein Objekt in Ihren Amazon-S3-Bucket hochladen kann. Die Verwendung einer vorsignierten URL ermöglicht einen Upload, ohne dass eine andere Partei über AWS Sicherheitsanmeldeinformationen oder Berechtigungen verfügen muss. Eine vorsignierte URL ist durch die Berechtigungen des Benutzers eingeschränkt, der sie erstellt hat. Das bedeutet, dass Sie ein Objekt, für das Sie eine vorsignierte URL zum Hochladen eines Objekts erhalten haben, nur dann hochladen können, wenn der Ersteller der URL die erforderlichen Berechtigungen zum Hochladen dieses Objekts besitzt.

Wenn jemand das Objekt über die URL hochlädt, erstellt Amazon S3 das Objekt in dem angegebenen Bucket. Wenn im Bucket bereits ein Objekt mit demselben Schlüssel vorhanden ist,

der in der vorsignierten URL angegeben wird, ersetzt Amazon S3 das vorhandene Objekt durch das hochgeladene Objekt. Nach dem Upload gehört das Objekt dem Bucket-Besitzer.

Allgemeine Informationen zu vorsignierten URLs finden Sie unter Arbeiten mit vorsignierten URLs.

Sie können mit AWS -Explorer für Visual Studio eine vorsignierte URL zum Hochladen eines Objekts erstellen, ohne einen Code schreiben zu müssen. Sie können eine vorsignierte URL mithilfe von AWS -SDKs auch programmgesteuert generieren.

Verwenden der AWS Toolkit for Visual Studio (Windows)

> **Note**
>
> Derzeit unterstützt Visual Studio für Mac AWS Toolkit for Visual Studio nicht.

1. Installieren Sie die AWS Toolkit for Visual Studio mit den folgenden Anweisungen: Installieren und Einrichten des Toolkit for Visual Studio im AWS Toolkit for Visual Studio -Benutzerhandbuch.
2. Stellen Sie eine Verbindung zu her, AWS indem Sie die folgenden Schritte ausführen: Herstellen einer Verbindung mit AWS im AWS Toolkit for Visual Studio -Benutzerhandbuch.
3. Klicken Sie im linken Bereich mit der Bezeichnung AWS Explorer mit der rechten Maustaste auf den Bucket, in den Sie ein Objekt hochladen möchten.
4. Wählen Sie Vorsignierte URL erstellen....
5. Legen Sie im Popup-Fenster das Ablaufdatum und die Ablaufzeit für Ihre vorsignierte URL fest.
6. Legen Sie für Objektschlüssel den Namen der Datei fest, die hochgeladen werden soll. Die Datei, die Sie hochladen, muss genau mit diesem Namen übereinstimmen. Wenn bereits ein Objekt mit demselben Objektschlüssel im Bucket vorhanden ist, ersetzt Amazon S3 das vorhandene Objekt durch das neu hochgeladene Objekt.
7. Wählen Sie PUT, um anzugeben, dass diese vorsignierte URL zum Hochladen eines Objekts verwendet wird.
8. Wählen Sie die Schaltfläche Generate (Generieren) aus.
9. Wählen Sie zum Kopieren der URL in die Zwischenablage Copy (Kopieren) aus.
10. Um diese URL zu verwenden, können Sie mit dem `curl`-Befehl eine PUT-Anfrage senden. Fügen Sie den vollständigen Pfad zu Ihrer Datei und die vorsignierte URL selbst ein.

```
curl -X PUT -T "/path/to/file" "presigned URL"
```

Verwenden der AWS SDKs

Beispiele für die Verwendung der - AWS SDKs zum Generieren einer vorsignierten URL zum Hochladen eines Objekts finden Sie unter Erstellen einer vorsignierten URL für Amazon S3 mithilfe eines - AWS SDK.

Wenn Sie die - AWS SDKs verwenden, um eine vorsignierte URL zu generieren, beträgt die maximale Ablaufzeit 7 Tage ab dem Zeitpunkt der Erstellung.

> **Note**
>
> Für alle , die nach dem 20. März 2019 AWS-Regionen gestartet wurden, müssen Sie die `endpoint-url` und `AWS-Region` mit der -Anforderung angeben. Eine Liste aller Amazon-S3-Regionen und Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS -Referenz.

Transformieren von Objekten mit S3 Object Lambda

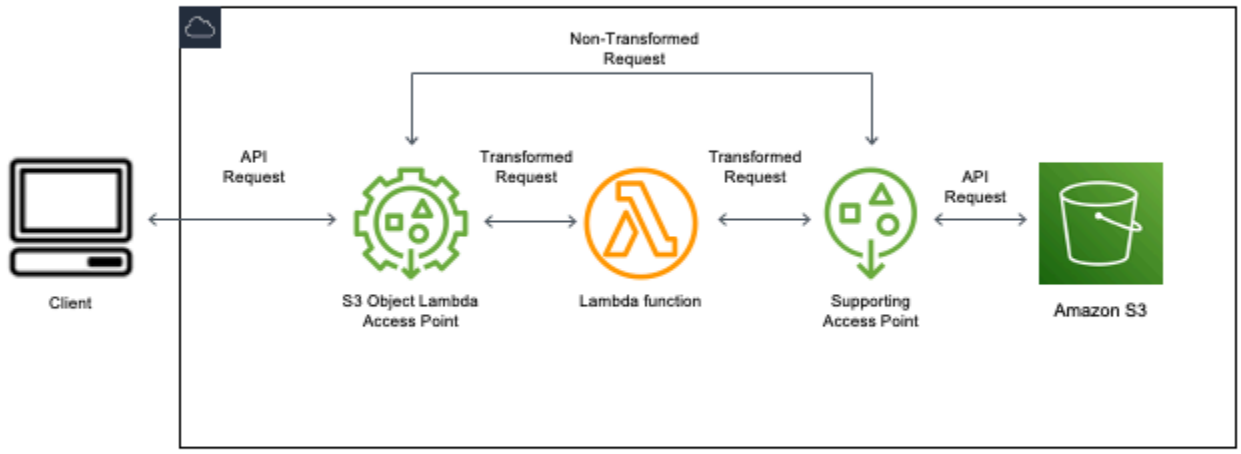
Mit S3 Object Lambda können Sie `GET`-, `LIST`- und `HEAD`-Anforderungen von Amazon S3 eigenen Code hinzufügen, um Daten zu ändern und zu verarbeiten, wenn sie an eine Anwendung zurückgegeben werden. Sie können mit benutzerdefiniertem Code die von S3-`GET`-Anforderungen zurückgegebenen Daten ändern, um Zeilen zu filtern, Bilder dynamisch in der Größe zu ändern und mit Wassermarken zu versehen, vertrauliche Daten zu redigieren und mehr. Sie können S3 Object Lambda auch verwenden, um die Ausgabe von S3-`LIST`-Anforderungen zu ändern und eine benutzerdefinierte Ansicht aller Objekte in einem Bucket zu erstellen, und von S3-`HEAD`-Anforderungen zum Ändern von Objektmetadaten wie Objektname und -größe. Sie können S3 Object Lambda als Ursprung für Ihre Amazon- CloudFront Verteilung verwenden, um Daten für Endbenutzer anzupassen, z. B. die Größe von Bildern automatisch anzupassen, ältere Formate zu transcodieren (z. B. von JPEG in WebP) oder Metadaten zu entfernen. Weitere Informationen finden Sie im AWS Blogbeitrag Verwenden von Amazon S3 Object Lambda mit Amazon CloudFront. Unterstützt von AWS Lambda-Funktionen wird Ihr Code auf einer Infrastruktur ausgeführt, die vollständig von verwaltet wird AWS. Mit S3 Object Lambda wird die Notwendigkeit reduziert, abgeleitete Kopien Ihrer Daten zu erstellen und zu speichern oder Proxys auszuführen. Dabei sind keine Änderungen an Ihren Anwendungen erforderlich.

So funktioniert S3 Object Lambda

S3 Object Lambda verwendet AWS Lambda Funktionen, um die Ausgabe von Standard-S3-`GET`, - `LIST`oder -`HEAD`Anforderungen automatisch zu verarbeiten. AWS Lambda ist ein Serverless-Rechenservice, der vom Kunden definierten Code ausführt, ohne dass die zugrunde liegenden Rechenressourcen verwaltet werden müssen. Sie können Ihre eigenen benutzerdefinierten Lambda-Funktionen erstellen und ausführen und die Datentransformation an Ihren spezifischen Anwendungsfall anpassen.

Nachdem Sie eine Lambda-Funktion konfiguriert haben, fügen Sie sie an einen Serviceendpunkt von S3 Object Lambda-an, der als Object Lambda Access Point bezeichnet wird. Der Object Lambda Access Point verwendet für den Zugriff auf Amazon S3 einen standardmäßigen S3-Zugriffspunkt, der als unterstützender Zugriffspunkt bezeichnet wird.

Wenn Sie eine Anforderung an Ihren Object Lambda Access Point senden, ruft Amazon S3 Ihre Lambda-Funktion automatisch auf. Dann geben alle Daten, die mit der S3-Anforderung `GET`, `LIST` oder `HEAD` über den Object Lambda Access Point abgerufen werden, ein transformiertes Ergebnis an die Anwendung zurück. Alle anderen Anforderungen werden wie gewohnt verarbeitet, wie im folgenden Diagramm dargestellt.



Die Themen in diesem Abschnitt beschreiben, wie Sie mit S3 Object Lambda arbeiten.

Themen

Erstellen von Objekt-Lambda-Zugriffspunkten

Ein Object Lambda Access Point ist mit genau einem Standard-Zugriffspunkt und folglich mit einem Amazon-S3-Bucket verknüpft. Zum Erstellen eines Object Lambda Access Point benötigen Sie folgende Ressourcen:

- Ein Amazon-S3-Bucket Informationen zum Erstellen von Buckets finden Sie unter the section called “Erstellen eines Buckets”.
- Ein standardmäßiger S3-Zugriffspunkt Wenn Sie mit Object Lambda Access Point arbeiten, wird dieser Standardzugriffspunkt als unterstützender Zugriffspunkt bezeichnet. Informationen zum Erstellen von Standardzugriffspunkten finden Sie unter the section called “Erstellen von Zugriffspunkten”.
- Eine - AWS Lambda Funktion. Sie können entweder Ihre eigene Lambda-Funktion erstellen oder eine vordefinierte Funktion verwenden. Weitere Informationen zum Erstellen von Lambda-Funktionen finden Sie unter the section called “Schreiben von Lambda-Funktionen”. Weitere Hinweise zu vorgefertigten Funktionen finden Sie unter Verwenden von AWS erstellten Lambda-Funktionen.
- (Optional) Eine AWS Identity and Access Management (IAM)-Richtlinie. Amazon-S3-Zugriffspunkte unterstützen IAM-Ressourcenrichtlinien, mit denen Sie die Verwendung des Zugriffspunkts nach Ressource, Benutzer oder anderen Bedingungen steuern können. Weitere Informationen zum Erstellen dieser Richtlinien finden Sie unter the section called “Konfigurieren von IAM-Richtlinien”.

In den folgenden Abschnitten wird beschrieben, wie Sie einen Object Lambda Access Point mit Folgendem erstellen:

- Die AWS Management Console
- Die AWS Command Line Interface (AWS CLI)
- Eine - AWS CloudFormation Vorlage
- Die AWS Cloud Development Kit (AWS CDK)

Informationen zum Erstellen eines Object Lambda Access Point mithilfe der REST-API finden Sie unter `CreateAccessPointForObjectLambda` in der API-Referenz zu Amazon Simple Storage Service.

Erstellen eines Object Lambda Access Point

Erstellen Sie Ihren Object Lambda Access Point mit einem der folgenden Verfahren.

Verwenden der S3-Konsole

So erstellen Sie einen Object Lambda Access Point mit der Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie auf der Seite Object Lambda access points (Objekt-Lambda-Zugriffspunkte) die Option Create Object Lambda access point (Objekt-Lambda-Zugriffspunkt erstellen).
4. Geben Sie unter Object Lambda access point name (Name des Objekt-Lambda-Zugriffspunkts) den Namen ein, den Sie für den Zugriffspunkt verwenden möchten.

   Wie bei Standard-Zugriffspunkten gibt es Regeln für die Benennung von Object Lambda Access Points. Weitere Informationen finden Sie unter Regeln zur Benennung von Amazon S3-Zugriffspunkten.
5. Geben Sie für Supporting Access Point (Unterstützender Access Point) den Standard-Zugriffspunkt ein, den Sie verwenden möchten, oder navigieren Sie zu diesem. Der Zugriffspunkt muss sich in derselben befinden AWS-Region wie die Objekte, die Sie transformieren möchten. Informationen zum Erstellen von Standardzugriffspunkten finden Sie unter the section called “Erstellen von Zugriffspunkten”.

6. Unter Transformationskonfiguration können Sie eine Funktion hinzufügen, die Ihre Daten für den Object Lambda Access Point transformiert. Führen Sie eine der folgenden Aktionen aus:

   - Wenn Sie bereits über eine AWS Lambda Funktion in Ihrem Konto verfügen, können Sie sie unter Lambda-Funktion aufrufen auswählen. Hier können Sie den Amazon-Ressourcennamen (ARN) einer Lambda-Funktion in Ihrem eingeben AWS-Konto oder eine Lambda-Funktion aus dem Dropdown-Menü auswählen.
   - Wenn Sie eine AWS erstellte Funktion verwenden möchten, wählen Sie den Funktionsnamen unter AWS der erstellten Funktion aus und wählen Sie Lambda-Funktion erstellen aus. Dadurch gelangen Sie zur Lambda-Konsole, in der Sie eine integrierte Funktion in Ihrem bereitstellen können AWS-Konto. Weitere Hinweise zu integrierten Funktionen finden Sie unter Verwenden von AWS erstellten Lambda-Funktionen.

   Wählen Sie unter S3 APIs (S3-APIs) eine oder mehrere API-Operationen aus, die aufgerufen werden sollen. Für jede ausgewählte API müssen Sie eine Lambda-Funktion angeben, die aufgerufen werden soll.

7. (Optional) Fügen Sie unter Payload (Nutzlast) JSON-Text hinzu, den Sie Ihrer Lambda-Funktion als Eingabe zur Verfügung stellen möchten. Sie können Nutzlasten mit verschiedenen Parametern für verschiedene Object Lambda Access Points konfigurieren, die dieselbe Lambda-Funktion aufrufen, wodurch die Flexibilität Ihrer Lambda-Funktion erhöht wird.

   > **Important**
   >
   > Wenn Sie Object Lambda Access Points verwenden, stellen Sie sicher, dass die Nutzlast keine vertraulichen Informationen enthält.

8. (Optional) Für Range and part number (Bereichs- und Teilenummer) müssen Sie diese Option aktivieren, wenn Sie `GET`- und `HEAD`-Anforderungen mit Bereichs- und Teilenummern-Headern bearbeiten möchten. Durch Aktivieren dieser Option wird bestätigt, dass Ihre Lambda-Funktion diese Anforderungen erkennen und bearbeiten kann. Weitere Informationen zu Bereichsüberschriften und Teilenummern finden Sie unter Arbeiten mit Range- und partNumber-Headern.

9. (Optional) Wählen Sie für Anforderungsmetriken die Option Aktivieren oder Deaktivieren aus, um Ihrem Object Lambda Access Point Amazon-S3-Überwachung hinzuzufügen. Anforderungsmetriken werden zum Amazon- CloudWatchStandardtarif abgerechnet.

10. (Optional) Legen Sie unter Object Lambda Access Point policy (Objekt-Lambda-Access-Point-Richtlinie) eine Ressourcenrichtlinie fest. Ressourcenrichtlinien erteilen Berechtigungen für den angegebenen Object Lambda Access Point und können die Verwendung des Zugriffspunkts nach Ressource, Benutzer oder anderen Bedingungen steuern. Weitere Hinweise zu Ressourcenrichtlinien für Objekt-Lambda-Zugriffspunkten finden Sie unter Konfigurieren von IAM-Richtlinien für Object Lambda Access Points.
11. Wählen Sie unter Block Public Access settings for this Object Lambda Access Point (Einstellungen für die Blockierung des öffentlichen Zugriffs für diesen Objekt-Lambda-Zugriffspunkt) die Einstellungen für die Blockierung des öffentlichen Zugriffs aus, die Sie anwenden möchten. Alle Einstellungen für die Blockierung des öffentlichen Zugriffs sind standardmäßig für neue Objekt-Lambda-Zugriffspunkte aktiviert. Wir empfehlen Ihnen, die Standardeinstellungen aktiviert zu lassen. Amazon S3 unterstützt derzeit nicht das Ändern der Einstellungen für die Blockierung des öffentlichen Zugriffs eines Objekt-Lambda-Zugriffspunkts, nachdem der Objekt-Lambda-Zugriffspunkt erstellt wurde.

    Weitere Informationen über die Blockierung des öffentlichen Zugriffs in Amazon S3 finden Sie unter Verwalten des öffentlichen Zugriffs auf Zugriffspunkte.
12. Wählen Sie Create Object Lambda Access Point (Objekt-Lambda-Access-Point erstellen) aus.

Verwenden der AWS CLI

So erstellen Sie einen Object Lambda Access Point mithilfe einer - AWS CloudFormation Vorlage

> **Note**
>
> Wenn Sie die folgenden Befehle verwenden, ersetzen Sie *`user input placeholders`* durch eigene Informationen.

1. Laden Sie das AWS Lambda Funktionsbereitstellungspaket `s3objectlambda_deployment_package.zip` unter S3 Object Lambda Standardkonfiguration herunter.
2. Führen Sie den folgenden `put-object`-Befehl aus, um das Paket in einen Amazon-S3-Bucket hochzuladen.

```
aws s3api put-object --bucket Amazon S3 bucket name --key
 s3objectlambda_deployment_package.zip --body release/
s3objectlambda_deployment_package.zip
```

3. Laden Sie die AWS CloudFormation Vorlage `s3objectlambda_defaultconfig.yaml` unter S3-Objekt-Lambda-Standardkonfiguration herunter.
4. Führen Sie den folgenden `deploy`-Befehl aus, um die Vorlage in Ihrem AWS-Konto bereitzustellen.

```
aws cloudformation deploy --template-file s3objectlambda_defaultconfig.yaml \
 --stack-name AWS CloudFormation stack name \
 --parameter-overrides ObjectLambdaAccessPointName=Object Lambda Access Point name
 \
  SupportingAccessPointName=Amazon S3 access point S3BucketName=Amazon S3 bucket \
  LambdaFunctionS3BucketName=Amazon S3 bucket containing your Lambda package \
  LambdaFunctionS3Key=Lambda object key LambdaFunctionS3ObjectVersion=Lambda object
 version \
  LambdaFunctionRuntime=Lambda function runtime --capabilities capability_IAM
```

Sie können diese AWS CloudFormation Vorlage so konfigurieren, dass Lambda für `GET-HEAD`, - und `LIST`-API-Operationen aufgerufen wird. Weitere Informationen zum Ändern der Standardkonfiguration der Vorlage finden Sie unter the section called “Automatisieren der S3-Objekt-Lambda-Einrichtung mit AWS CloudFormation”.

So erstellen Sie einen Object Lambda Access Point mithilfe der AWS CLI

> **Note**
>
> Wenn Sie die folgenden Befehle verwenden, ersetzen Sie *`user input placeholders`* durch eigene Informationen.

Im folgenden Beispiel wird ein Object Lambda Access Point mit dem Namen *`my-object-lambda-ap`* für den Bucket *`DOC-EXAMPLE-BUCKET1`* im Konto *`111122223333`* erstellt. In diesem Beispiel wird davon ausgegangen, dass bereits ein Standard-Zugriffspunkt mit dem Namen *`example-ap`* erstellt wurde. Informationen zum Erstellen eines Standardzugriffspunkts finden Sie unter the section called “Erstellen von Zugriffspunkten”.

In diesem Beispiel wird die AWS vorgefertigte Funktion verwendet`decompress`. Weitere Hinweise zu vorgefertigten Funktionen finden Sie unter the section called “Verwenden von AWS erstellten Funktionen”.

1. Erstellen Sie einen Bucket. In diesem Beispiel verwenden wir *`DOC-EXAMPLE-BUCKET1`*. Informationen zum Erstellen von Buckets finden Sie unter the section called “Erstellen eines Buckets”.
2. Erstellen Sie einen Standardzugriffspunkt und hängen Sie ihn an Ihren Bucket an. In diesem Beispiel verwenden wir *`example-ap`*. Informationen zum Erstellen von Standardzugriffspunkten finden Sie unter the section called “Erstellen von Zugriffspunkten”.
3. Führen Sie eine der folgenden Aktionen aus:

   - Erstellen Sie in Ihrem Konto eine Lambda-Funktion, mit der Sie Ihr Amazon-S3-Objekt transformieren möchten. Weitere Informationen zum Erstellen von Lambda-Funktionen finden Sie unter the section called “Schreiben von Lambda-Funktionen”. Informationen zur Verwendung Ihrer benutzerdefinierten Funktion mit der AWS CLI finden Sie unter Verwenden von Lambda mit der AWS CLI im AWS Lambda -Entwicklerhandbuch.
   - Verwenden Sie eine AWS vorgefertigte Lambda-Funktion. Weitere Hinweise zu vorgefertigten Funktionen finden Sie unter Verwenden von AWS erstellten Lambda-Funktionen.
4. Erstellen einer JSON-Konfigurationsdate namens `my-olap-configuration.json`. Geben Sie in dieser Konfiguration den unterstützenden Zugriffspunkt und den Amazon-Ressourcennamen (ARN) für die Lambda-Funktion an, die Sie in den vorherigen Schritten erstellt haben, oder den ARN für die vorkonfigurierte Funktion, die Sie verwenden.

   Example

```
{
    "SupportingAccessPoint" : "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
    "TransformationConfigurations": [{
        "Actions" : ["GetObject", "HeadObject", "ListObjects", "ListObjectsV2"],
        "ContentTransformation" : {
            "AwsLambda": {
                "FunctionPayload" : "{\"compressionType\":\"gzip\"}",
                "FunctionArn" : "arn:aws:lambda:us-east-1:111122223333:function/
compress"
            }
        }
```

```
    }]
}
```

5. Führen Sie den Befehl `create-access-point-for-object-lambda` aus, um den Object Lambda Access Point zu erstellen.

```
aws s3control create-access-point-for-object-lambda --account-id 111122223333 --
name my-object-lambda-ap --configuration file://my-olap-configuration.json
```

6. (Optional) Erstellen Sie eine JSON-Richtliniendatei namens `my-olap-policy.json`.

   Durch Hinzufügen einer Ressourcenrichtlinie für Objekt-Lambda-Zugriffspunkte lässt sich die Verwendung des Zugriffspunkts nach Ressource, Benutzer oder anderen Bedingungen steuern. Diese Ressourcenrichtlinie gewährt dem angegebenen Object Lambda Access Point die `GetObject`-Berechtigung für das Konto *444455556666*.

   Example

```
{
    "Version": "2008-10-17",
    "Statement": [
        {
            "Sid": "Grant account 444455556666 GetObject access",
            "Effect": "Allow",
            "Action": "s3-object-lambda:GetObject",
            "Principal": {
                "AWS": "arn:aws:iam::444455556666:root"
            },
            "Resource": "your-object-lambda-access-point-arn"
        }
    ]
}
```

7. (Optional) Führen Sie den Befehl `put-access-point-policy-for-object-lambda` aus, um Ihre Ressourcenrichtlinie festzulegen.

```
aws s3control put-access-point-policy-for-object-lambda --account-id 111122223333
 --name my-object-lambda-ap --policy file://my-olap-policy.json
```

8. (Optional) Geben Sie eine Nutzlast an.

Eine Nutzlast ist ein optionales JSON, das Sie Ihrer AWS Lambda Funktion als Eingabe zur Verfügung stellen können. Sie können Nutzlasten mit verschiedenen Parametern für verschiedene Object Lambda Access Points konfigurieren, die dieselbe Lambda-Funktion aufrufen, wodurch die Flexibilität Ihrer Lambda-Funktion erhöht wird.

Die folgende Konfiguration des Object Lambda Access Point zeigt eine Nutzlast mit zwei Parametern.

```
{
 "SupportingAccessPoint": "AccessPointArn",
 "CloudWatchMetricsEnabled": false,
 "TransformationConfigurations": [{
  "Actions": ["GetObject", "HeadObject", "ListObjects", "ListObjectsV2"],
  "ContentTransformation": {
   "AwsLambda": {
    "FunctionArn": "FunctionArn",
    "FunctionPayload": "{\"res-x\": \"100\",\"res-y\": \"100\"}"
   }
  }
 }]
}
```

Die folgende Konfiguration des Object Lambda Access Point zeigt eine Nutzlast mit einem Parameter und den aktivierten Einstellungen `GetObject-Range`, `GetObject-PartNumber`, `HeadObject-Range` und `HeadObject-PartNumber`.

```
{
    "SupportingAccessPoint":"AccessPointArn",
    "CloudWatchMetricsEnabled": false,
    "AllowedFeatures": ["GetObject-Range", "GetObject-PartNumber", "HeadObject-
Range", "HeadObject-PartNumber"],
    "TransformationConfigurations": [{
        "Action": ["GetObject", "HeadObject", "ListObjects", "ListObjectsV2"],
        "ContentTransformation": {
            "AwsLambda": {
                "FunctionArn":"FunctionArn",
                "FunctionPayload": "{\"compression-amount\": \"5\"}"
            }
        }
    }]
```

```
}
```

> **Important**
>
> Wenn Sie Object Lambda Access Points verwenden, stellen Sie sicher, dass die Nutzlast keine vertraulichen Informationen enthält.

Verwenden der AWS CloudFormation Konsole und Vorlage

Sie können einen Object Lambda Access Point mithilfe der von Amazon S3 bereitgestellten Standardkonfiguration erstellen. Sie können eine - AWS CloudFormation Vorlage und einen Lambda-Funktionsquellcode aus dem GitHub Repository herunterladen und diese Ressourcen bereitstellen, um einen funktionalen Object Lambda Access Point einzurichten.

Informationen zum Ändern der Standardkonfiguration der AWS CloudFormation Vorlage finden Sie unter the section called “Automatisieren der S3-Objekt-Lambda-Einrichtung mit AWS CloudFormation”.

Informationen zum Konfigurieren von Object Lambda Access Points mithilfe von AWS CloudFormation ohne die Vorlage finden Sie unter `AWS::S3ObjectLambda::AccessPoint` im AWS CloudFormation -Benutzerhandbuch.

So laden Sie das Bereitstellungspaket für die Lambda-Funktion hoch

1. Laden Sie das AWS Lambda Funktionsbereitstellungspaket `s3objectlambda_deployment_package.zip` unter S3 Object Lambda Standardkonfiguration herunter.
2. Hochladen des Pakets in einen Amazon S3 Bucket

So erstellen Sie einen Object Lambda Access Point mithilfe der AWS CloudFormation Konsole

1. Laden Sie die AWS CloudFormation Vorlage `s3objectlambda_defaultconfig.yaml` unter S3 Object Lambda Standardkonfiguration herunter.
2. Melden Sie sich bei der - AWS Managementkonsole an und öffnen Sie die - AWS CloudFormation Konsole unter https://console.aws.amazon.com/cloudformation.
3. Führen Sie eine der folgenden Aktionen aus:

- Wenn Sie AWS CloudFormation noch nie verwendet haben, wählen Sie auf der - AWS CloudFormation Startseite Stack erstellen aus.
- Wenn Sie AWS CloudFormation bereits verwendet haben, wählen Sie im linken Navigationsbereich Stacks aus. Wählen Sie Create stack (Stack erstellen) und dann With new resources (standard) (Mit neuen Ressourcen (Standard)) aus.

4. Wählen Sie für Voraussetzung - Vorlage vorbereiten die Option Vorlage ist bereit aus.
5. Wählen Sie unter Specify template (Vorlage angeben) die Option Upload a template file (Vorlagendatei hochladen) aus und laden Sie `s3objectlambda_defaultconfig.yaml` hoch.
6. Wählen Sie Weiter aus.
7. Geben Sie auf der Seite Stackdetails angeben einen Namen für den Stack ein.
8. Geben Sie im Abschnitt Parameters (Parameter) die folgenden Parameter an, die in der Stack-Vorlage definiert sind:

   a. Führen Sie für CreateNewSupportingAccessPoint einen der folgenden Schritte aus:

      - Wenn Sie bereits einen unterstützenden Zugriffspunkt für den S3-Bucket haben, in den Sie die Vorlage hochgeladen haben, wählen Sie false (falsch) aus.
      - Wenn Sie einen neuen Zugriffspunkt für diesen Bucket erstellen möchten, wählen Sie true (wahr) aus.

   b. EnableCloudWatchMonitoringWählen Sie für true oder false aus, je nachdem, ob Sie Amazon- CloudWatch Anforderungsmetriken und Alarme aktivieren möchten.

   c. (Optional) Fügen Sie für JSON-Text LambdaFunctionPayloadhinzu, den Sie Ihrer Lambda-Funktion als Eingabe bereitstellen möchten. Sie können Nutzlasten mit verschiedenen Parametern für verschiedene Object Lambda Access Points konfigurieren, die dieselbe Lambda-Funktion aufrufen, wodurch die Flexibilität Ihrer Lambda-Funktion erhöht wird.

      > **Important**
      >
      > Wenn Sie Object Lambda Access Points verwenden, stellen Sie sicher, dass die Nutzlast keine vertraulichen Informationen enthält.

   d. LambdaFunctionRuntimeGeben Sie für Ihre bevorzugte Laufzeit für die Lambda-Funktion ein. Die verfügbaren Auswahlmöglichkeiten sind `nodejs14.x`, `python3.9`, `java11`.

   e. Geben Sie für LambdaFunctionS3BucketName den Namen des Amazon S3-Buckets ein, in den Sie das Bereitstellungspaket hochgeladen haben.

f. Geben Sie für LambdaFunctionS3Key den Amazon S3-Objektschlüssel ein, in den Sie das Bereitstellungspaket hochgeladen haben.

g. Geben Sie für LambdaFunctionS3ObjectVersion die Amazon S3-Objektversion ein, in die Sie das Bereitstellungspaket hochgeladen haben.

h. ObjectLambdaAccessPointNameGeben Sie für einen Namen für Ihren Object Lambda Access Point ein.

i. Geben Sie für S3BucketName den Namen des Amazon S3-Buckets ein, der Ihrem Object Lambda Access Point zugeordnet werden soll.

j. SupportingAccessPointNameGeben Sie für den Namen Ihres unterstützenden Zugriffspunkts ein.

> **Note**
>
> Dies ist ein Zugriffspunkt, der mit dem Amazon-S3-Bucket verknüpft ist, den Sie im vorherigen Schritt ausgewählt haben. Wenn Ihrem Amazon S3-Bucket keine Zugriffspunkte zugeordnet sind, können Sie die Vorlage so konfigurieren, dass sie einen für Sie erstellt, indem Sie für die Option „true“ auswählenCreateNewSupportingAccessPoint.

9. Wählen Sie Weiter aus.
10. Wählen Sie auf der Seite Configure stack options (Stack-Optionen konfigurieren) Next (Weiter) aus.

    Weitere Informationen zu den optionalen Einstellungen auf dieser Seite finden Sie unter Einstellen von AWS CloudFormation -Stack-Optionen im AWS CloudFormation -Benutzerhandbuch.
11. Wählen Sie auf der Seite Prüfen Stack erstellen aus.

Verwenden der AWS Cloud Development Kit (AWS CDK)

Weitere Informationen zum Konfigurieren von Object Lambda Access Points mithilfe der finden Sie AWS CDK unter `AWS::S3ObjectLambda` Construct Library in der APIAWS Cloud Development Kit (AWS CDK) -Referenz zu .

Automatisieren der Einrichtung von S3 Object Lambda mit einer CloudFormation Vorlage

Sie können eine - AWS CloudFormation Vorlage verwenden, um schnell einen Amazon S3 Object Lambda Access Point zu erstellen. Die CloudFormation Vorlage erstellt automatisch relevante Ressourcen, konfiguriert AWS Identity and Access Management (IAM)-Rollen und richtet eine - AWS Lambda Funktion ein, die Anfragen automatisch über den Object Lambda Access Point verarbeitet. Mit der CloudFormation Vorlage können Sie bewährte Methoden implementieren, Ihren Sicherheitsstatus verbessern und Fehler reduzieren, die durch manuelle Prozesse verursacht werden.

Dieses GitHub Repository enthält die CloudFormation Vorlage und den Quellcode der Lambda-Funktion. Anweisungen zur Verwendung der Vorlage finden Sie unter the section called “Erstellen von Objekt-Lambda-Zugriffspunkten”.

Die in der Vorlage bereitgestellte Lambda-Funktion führt keine Transformation aus. Stattdessen gibt sie Ihre Objekte im Ist-Zustand aus Ihrem S3-Bucket zurück. Sie können die Funktion klonen und Ihren eigenen Transformationscode hinzufügen, um Daten zu ändern und zu verarbeiten, wenn sie an eine Anwendung zurückgegeben werden. Weitere Informationen zum Anpassen Ihrer Funktion finden Sie unter the section called “Ändern der Lambda-Funktion” und the section called “Schreiben von Lambda-Funktionen”.

Ändern der Vorlage

Erstellen eines neuen unterstützenden Zugriffspunkts

S3 Object Lambda verwendet zwei Zugriffspunkte, einen Object Lambda Access Point und einen standardmäßigen S3-Zugriffspunkt, der als unterstützender Zugriffspunkt bezeichnet wird. Wenn Sie eine Anforderung an einen Object Lambda Access Point ausführen, ruft S3 entweder Lambda in Ihrem Namen auf oder delegiert die Anforderung an den unterstützenden Zugriffspunkt, abhängig von der Konfiguration von S3 Object Lambda. Sie können einen neuen unterstützenden Zugriffspunkt erstellen, indem Sie den folgenden Parameter als Teil des `aws cloudformation deploy`-Befehls, der beim Bereitstellen der Vorlage erstellt werden soll, übergeben.

```
CreateNewSupportingAccessPoint=true
```

Konfigurieren einer Nutzlast-Funktion

Sie können eine Nutzlast konfigurieren, um der Lambda-Funktion zusätzliche Daten bereitzustellen, indem Sie beim Bereitstellen der Vorlage den folgenden Parameter als Teil des `aws cloudformation deploy`-Befehls übergeben.

```
LambdaFunctionPayload="format=json"
```

Aktivieren der Amazon- CloudWatch Überwachung

Sie können die CloudWatch Überwachung aktivieren, indem Sie bei der Bereitstellung der Vorlage den folgenden Parameter als Teil des `aws cloudformation deploy` Befehls übergeben.

```
EnableCloudWatchMonitoring=true
```

Dieser Parameter aktiviert Ihren Object Lambda Access Point für Amazon S3-Anforderungsmetriken und erstellt zwei CloudWatch Alarme, um clientseitige und serverseitige Fehler zu überwachen.

> **Note**
>
> Für die Amazon- CloudWatch Nutzung fallen zusätzliche Kosten an. Weitere Informationen zu Amazon-S3-Anforderungsmetriken finden Sie unter Überwachen und Protokollieren von Zugriffspunkten.
> Details zu den Preisen finden Sie unter CloudWatch -Preise.

Konfigurieren von Provisioned Concurrency

Zur Reduzierung der Latenz können Sie bereitgestellte Parallelität für die Lambda-Funktion konfigurieren, die den Object Lambda Access Point unterstützt, indem Sie die Vorlage so bearbeiten, dass sie die folgenden Zeilen unter `Resources` einschließt.

```
LambdaFunctionVersion:
      Type: AWS::Lambda::Version
      Properties:
        FunctionName: !Ref LambdaFunction
        ProvisionedConcurrencyConfig:
            ProvisionedConcurrentExecutions: Integer
```

> **Note**
>
> Für die Bereitstellung von Nebenläufigkeit fallen zusätzliche Gebühren an. Weitere Informationen zur bereitgestellten Parallelität finden Sie unter Verwalten der von Lambda bereitgestellten Parallelität im AWS Lambda -Entwicklerhandbuch.
> Details zu den Preisen finden Sie unter AWS Lambda -Preise.

Ändern der Lambda-Funktion

Ändern der Header-Werte für eine **`GetObject`**-Anforderung

Standardmäßig leitet die Lambda-Funktion alle Header mit Ausnahme von `Content-Length` und `ETag` von der vorsignierten URL-Anforderung an den `GetObject`-Client weiter. Basierend auf Ihrem Transformationscode in der Lambda-Funktion können Sie wählen, ob Sie neue Header-Werte an den `GetObject`-Client senden.

Sie können Ihre Lambda-Funktion aktualisieren, um neue Header-Werte zu senden, indem Sie sie in der `WriteGetObjectResponse`-API-Operation übergeben.

Wenn Ihre Lambda-Funktion beispielsweise den Text in Amazon-S3-Objekten in eine andere Sprache übersetzt, können Sie einen neuen Wert im `Content-Language`-Header bestätigen. Ändern Sie die `writeResponse`-Funktion dazu wie folgt:

```
async function writeResponse (s3Client: S3, requestContext: GetObjectContext,
 transformedObject: Buffer,
  headers: Headers): Promise<PromiseResult<{}, AWSError>> {
  const { algorithm, digest } = getChecksum(transformedObject);

  return s3Client.writeGetObjectResponse({
    RequestRoute: requestContext.outputRoute,
    RequestToken: requestContext.outputToken,
    Body: transformedObject,
    Metadata: {
      'body-checksum-algorithm': algorithm,
      'body-checksum-digest': digest
    },
    ...headers,
    ContentLanguage: 'my-new-language'
  }).promise();
}
```

Eine vollständige Liste der unterstützten Header finden Sie unter `WriteGetObjectResponse` in der API-Referenz zu Amazon Simple Storage Service.

Zurückgeben von Metadaten Headern

Sie können Ihre Lambda-Funktion aktualisieren, um neue Header-Werte zu senden, indem Sie sie in der Anforderung der API-Operation `WriteGetObjectResponse` übergeben.

```
async function writeResponse (s3Client: S3, requestContext: GetObjectContext,
 transformedObject: Buffer,
  headers: Headers): Promise<PromiseResult<{}, AWSError>> {
  const { algorithm, digest } = getChecksum(transformedObject);

  return s3Client.writeGetObjectResponse({
    RequestRoute: requestContext.outputRoute,
    RequestToken: requestContext.outputToken,
    Body: transformedObject,
    Metadata: {
      'body-checksum-algorithm': algorithm,
      'body-checksum-digest': digest,
      'my-new-header': 'my-new-value'
    },
    ...headers
  }).promise();
}
```

Einen neuen Statuscode zurückgeben

Sie können einen benutzerdefinierten Statuscode an den `GetObject`-Client zurückgeben, indem Sie ihn in der Anforderung der API-Operation `WriteGetObjectResponse` übergeben.

```
async function writeResponse (s3Client: S3, requestContext: GetObjectContext,
 transformedObject: Buffer,
  headers: Headers): Promise<PromiseResult<{}, AWSError>> {
  const { algorithm, digest } = getChecksum(transformedObject);

  return s3Client.writeGetObjectResponse({
    RequestRoute: requestContext.outputRoute,
    RequestToken: requestContext.outputToken,
    Body: transformedObject,
    Metadata: {
      'body-checksum-algorithm': algorithm,
```

```
      'body-checksum-digest': digest
    },
    ...headers,
    StatusCode: Integer
  }).promise();
}
```

Eine vollständige Liste der unterstützten Statuscodes finden Sie unter `WriteGetObjectResponse` in der API-Referenz zu Amazon Simple Storage Service.

Anwenden der Parameter **`Range`** und **`partNumber`** auf das Quellobjekt

Standardmäßig kann der von der CloudFormation Vorlage erstellte Object Lambda Access Point die `partNumber` Parameter `Range` und verarbeiten. Die Lambda-Funktion wendet den Bereich oder die Teilenummer an, die auf das transformierte Objekt angefordert wird. Dazu muss die Funktion das gesamte Objekt herunterladen und die Transformation ausführen. In einigen Fällen werden Ihre transformierten Objektbereiche möglicherweise genau Ihren Quellobjektbereichen zugeordnet. Dies bedeutet, dass das Anfordern des Bytebereichs A-B für Ihr Quellobjekt und das Ausführen der Transformation zum gleichen Ergebnis führen kann, wie das Anfordern des gesamten Objekts, das Ausführen der Transformation und das Zurückgeben des Bytebereichs A-B für das transformierte Objekt.

In solchen Fällen können Sie die Implementierung der Lambda-Funktion ändern, um den Bereich oder die Teilenummer direkt auf das Quellobjekt anzuwenden. Dieser Ansatz verbessert die allgemeine Funktionslatenz und den erforderlichen Speicher. Weitere Informationen finden Sie unter the section called “Arbeiten mit Range- und partNumber-Headern”.

Deaktivieren der Verarbeitung von **`Range`** und **`partNumber`**

Standardmäßig kann der von der CloudFormation Vorlage erstellte Object Lambda Access Point die `partNumber` Parameter `Range` und verarbeiten. Wenn Sie dieses Verhalten nicht benötigen, können Sie es deaktivieren, indem Sie die folgenden Zeilen aus der Vorlage entfernen.

```
AllowedFeatures:
  - GetObject-Range
  - GetObject-PartNumber
  - HeadObject-Range
  - HeadObject-PartNumber
```

Transformieren von großen Objekten

Standardmäßig verarbeitet die Lambda-Funktion das gesamte Objekt im Speicher, bevor es mit dem Streamen der Antwort auf S3 Object Lambda beginnen kann. Sie können die Funktion ändern, um die Antwort zu streamen, während sie die Transformation durchführt. Dies hilft, die Transformationslatenz und die Speichergröße der Lambda-Funktion zu reduzieren. Eine Beispielimplementierung finden Sie unter Beispiel für komprimierte Inhalte streamen.

Verwenden von Amazon S3 Objekt-Lambda-Zugriffspunkten

Sie können Anforderungen über Amazon S3 Object Lambda Access Points ebenso vornehmen wie über andere Zugriffspunkte. Weitere Informationen darüber, wie Sie Anforderungen über einen Zugriffspunkt vornehmen können, finden Sie unter Verwenden von Zugriffspunkten. Sie können Anforderungen über Object Lambda Access Points über die Amazon S3-Konsole, AWS Command Line Interface (AWS CLI), AWS SDKs oder die Amazon S3-REST-API stellen.

> **Important**
>
> Die Amazon-Ressourcennamen (ARNs) für Object Lambda Access Points verwenden den Servicenamen `s3-object-lambda`. Folglich beginnen die ARNs von Object Lambda Access Points mit `arn:aws::s3-object-lambda` und nicht mit `arn:aws::s3`, was andere Zugriffspunkte verwenden.

So finden Sie den ARN für Ihren Object Lambda Access Point

Um einen Object Lambda Access Point mit der AWS CLI oder AWS SDKs zu verwenden, müssen Sie den Amazon-Ressourcennamen (ARN) des Object Lambda Access Point kennen. Die folgenden Beispiele zeigen, wie Sie den ARN für einen Object Lambda Access Point mithilfe der Amazon-S3-Konsole oder der AWS CLI finden.

Verwenden der S3-Konsole

So finden Sie den ARN für Ihren Object Lambda Access Point mithilfe der Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Wählen Sie das Optionsfeld neben dem Object Lambda Access Point aus, dessen ARN Sie kopieren möchten.

4. Klicken Sie auf Copy ARN (ARN kopieren).

Verwenden der AWS CLI

So finden Sie den ARN für Ihren Object Lambda Access Point mithilfe der AWS CLI

1. Führen Sie den folgenden Befehl aus, um eine Liste der Object Lambda Access Points abzurufen, die mit Ihrem AWS-Konto verbunden sind. Bevor Sie den Befehl ausführen, ersetzen Sie die Konto-ID *111122223333* durch Ihre AWS-Konto -ID.

```
aws s3control list-access-points-for-object-lambda --account-id 111122223333
```

2. Überprüfen Sie die Befehlsausgabe, um den ARN des Object Lambda Access Point zu finden, den Sie verwenden möchten. Die Ausgabe des vorherigen Befehls sollte dem folgenden Beispiel gleichen.

```
{
    "ObjectLambdaAccessPointList": [
        {
            "Name": "my-object-lambda-ap",
            "ObjectLambdaAccessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/my-object-lambda-ap"
        },
        ...
    ]
}
```

So verwenden Sie einen Alias im Bucket-Stil für den Object Lambda Access Point Ihres S3-Buckets

Wenn Sie einen Object Lambda Access Point erstellen, generiert Amazon S3 automatisch einen eindeutigen Alias für den Object Lambda Access Point. Sie können diesen Alias anstelle eines Amazon-S3-Bucketnamens oder des Amazon-Ressourcennamens (ARN) des Object Lambda Access Point in einer Anforderung für Zugriffspunkt-Operationen auf Datenebene verwenden. Eine Liste dieser Vorgänge finden Sie unter Kompatibilität von Zugriffspunkten mit - AWS Services.

Ein Aliasname eines Object Lambda Access Point wird innerhalb desselben Namespace wie ein Amazon-S3-Bucket erstellt. Dieser Aliasname wird automatisch generiert und kann nicht geändert werden. Für einen vorhandenen Object Lambda Access Point wird automatisch ein

Alias zur Verwendung zugewiesen. Ein Aliasname eines Object Lambda Access Point erfüllt alle Anforderungen eines gültigen Amazon-S3-Bucketnamens und besteht aus den folgenden Teilen:

Object Lambda Access Point name prefix-*metadata*--ol-s3

> **Note**
>
> Das Suffix `--ol-s3` ist für Aliasnamen von Object Lambda Access Points reserviert und kann nicht für die Bucket-Namen oder die Namen von Object Lambda Access Points verwendet werden. Weitere Informationen zu Amazon-S3-Bucket-Benennungsregeln finden Sie unter Regeln für die Benennung von Buckets.

Das folgende Beispiel zeigt den ARN und den Alias für einen Object Lambda Access Point namens *my-object-lambda-access-point*:

- ARN – arn:aws:s3-object-lambda:*region*:*account-id*:accesspoint/*my-object-lambda-access-point*
- Alias des Object Lambda Access Point – *my-object-lambda-acc*-*1a4n8yjrb3kda96f67zwrwiiuse1a*--ol-s3

Wenn Sie einen Object Lambda Access Point verwenden, können Sie den Aliasnamen des Object Lambda Access Point verwenden, ohne dass umfangreiche Codeänderungen erforderlich sind.

Wenn Sie einen Object Lambda Access Point löschen, wird der Aliasname des Object Lambda Access Point inaktiv und die Bereitstellung wird aufgehoben.

So finden Sie den Alias für Ihren Object Lambda Access Point

Verwenden der S3-Konsole

So finden Sie den Alias für Ihren Object Lambda Access Point mithilfe der Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Objekt-Lambda-Zugriffspunkte aus.
3. Kopieren Sie den Wert unter Alias des Objekt-Lambda-Zugriffspunkts für den Object Lambda Access Point, den Sie verwenden möchten.

Verwenden der AWS CLI

Wenn Sie einen Object Lambda Access Point erstellen, generiert Amazon S3 automatisch einen Aliasnamen für den Object Lambda Access Point wie im folgenden Beispielbefehl gezeigt. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen. Informationen zum Erstellen eines Object Lambda Access Point mithilfe der finden Sie AWS CLI unter So erstellen Sie einen Object Lambda Access Point mithilfe der AWS CLI.

```
aws s3control create-access-point-for-object-lambda --account-id 111122223333 --
name my-object-lambda-access-point --configuration file://my-olap-configuration.json
{
    "ObjectLambdaAccessPointArn": "arn:aws:s3:region:111122223333:accesspoint/my-
access-point",
    "Alias": {
        "Value": "my-object-lambda-acc-1a4n8yjrb3kda96f67zwrwiiuse1a--ol-s3",
        "Status": "READY"
    }
}
```

Der generierte Aliasname des Object Lambda Access Point besteht aus zwei Feldern:

- Das Feld `Value` ist der Aliaswert des Object Lambda Access Point.
- Das Feld `Status` ist der Status des Alias des Object Lambda Access Point. Wenn der Status `PROVISIONING` lautet, stellt Amazon S3 den Alias des Object Lambda Access Point bereit und der Alias kann noch nicht verwendet werden. Wenn der Status `READY` lautet, wurde der Alias des Object Lambda Access Point erfolgreich bereitgestellt und kann verwendet werden.

Weitere Informationen zum Datentyp `ObjectLambdaAccessPointAlias` in der REST-API finden Sie unter CreateAccessPointForObjectLambda und ObjectLambdaAccessPointAlias in der API-Referenz zu Amazon Simple Storage Service.

So verwenden Sie den Alias des Object Lambda Access Point

Sie können einen Alias eines Object Lambda Access Point anstelle eines Amazon-S3-Bucketnamens für die unter Kompatibilität von Zugriffspunkten mit - AWS Services aufgeführten Operationen verwenden.

Im folgenden AWS CLI Beispiel für den `get-bucket-location` Befehl wird der Zugriffspunkt-Alias des Buckets verwendet, um die zurückzugeben AWS-Region , in der sich der Bucket befindet. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3api get-bucket-location --bucket my-object-lambda-
acc-w7i37nq6xuzgax3jw3oqtifiusw2a--ol-s3

{
    "LocationConstraint": "us-west-2"
}
```

Wenn der Alias des Object Lambda Access Point in einer Anforderung nicht gültig ist, wird der Fehlercode `InvalidAccessPointAliasError` zurückgegeben. Weitere Informationen zu `InvalidAccessPointAliasError` finden Sie unter Liste der Fehlercodes in der API-Referenz zu Amazon Simple Storage Service.

Die Einschränkungen eines Alias eines Object Lambda Access Point sind die gleichen wie bei einem Zugriffspunkt-Alias. Weitere Informationen zu den Einschränkungen eines Zugriffspunkt-Alias finden Sie unter Einschränkungen.

Sicherheitsüberlegungen für S3 Object Lambda-Zugriffspunkte

Mit Amazon S3 Object Lambda können Sie benutzerdefinierte Transformationen an Daten durchführen, während es Amazon S3 verlässt, indem Sie die Skalierung und Flexibilität von AWS Lambda als Datenverarbeitungsplattform verwenden. S3 und Lambda bleiben standardmäßig sicher, aber besondere Rücksicht vom Lambda-Funktionsautor ist erforderlich, um diese Sicherheit zu gewährleisten. S3 Object Lambda verlangt, dass der gesamte Zugriff von authentifizierten Prinzipalen (kein anonymer Zugriff) und über HTTPS erfolgt.

Es wird Folgendes empfohlen, um Sicherheitsrisiken zu minimieren:

- Beschränken Sie die Lambda-Ausführungsrolle auf den geringsten Satz von Berechtigungen.
- Stellen Sie nach Möglichkeit sicher, dass Ihre Lambda-Funktion über die bereitgestellte vorsignierte URL auf Amazon S3 zugreift.

Konfigurieren von IAM-Richtlinien

S3-Zugriffspunkte unterstützen AWS Identity and Access Management (IAM)-Ressourcenrichtlinien, mit denen Sie die Verwendung des Zugriffspunkts nach Ressource, Benutzer oder anderen Bedingungen steuern können. Weitere Informationen finden Sie unter Konfigurieren von IAM-Richtlinien für Object Lambda Access Points.

Verhalten der Verschlüsselung

Da Object Lambda Access Points sowohl Amazon S3 als auch verwenden AWS Lambda, gibt es Unterschiede im Verschlüsselungsverhalten. Weitere Informationen zum standardmäßigen S3-Verschlüsselungsverhalten finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

- Wenn Sie die serverseitige Verschlüsselung in S3 mit Object Lambda Access Points verwenden, wird das Objekt entschlüsselt, bevor es an Lambda gesendet wird. Nachdem das Objekt an Lambda gesendet wurde, wird es unverschlüsselt verarbeitet (im Fall einer `GET`- oder `HEAD`-Anforderung).
- Um zu verhindern, dass der Verschlüsselungsschlüssel protokolliert wird, lehnt S3 `GET`- und `HEAD`-Anforderungen für Objekte ab, die mit vom Kunden bereitgestellte Schlüssel über serverseitige Verschlüsselung (SSE-C) verschlüsselt wurden. Die Lambda-Funktion kann diese Objekte jedoch weiterhin abrufen, sofern sie Zugriff auf den vom Client bereitgestellten Schlüssel hat.
- Wenn Sie die clientseitige Verschlüsselung in S3 mit Object Lambda Access Points verwenden, stellen Sie sicher, dass Lambda Zugriff auf den Verschlüsselungsschlüssel hat, damit es das Objekt entschlüsseln und erneut verschlüsseln kann.

Sicherheit für Zugriffspunkte

S3 Object Lambda verwendet zwei Zugriffspunkte, einen Object Lambda Access Point und einen standardmäßigen S3-Zugriffspunkt, der als unterstützender Zugriffspunkt bezeichnet wird. Wenn Sie eine Anforderung an einen Object Lambda Access Point ausführen, ruft S3 entweder Lambda in Ihrem Namen auf oder delegiert die Anforderung an den unterstützenden Zugriffspunkt, abhängig von der Konfiguration von S3 Object Lambda. Wenn Lambda für eine Anforderung aufgerufen wird, generiert S3 eine vorsignierte URL zu Ihrem Objekt in Ihrem Namen über den unterstützenden Zugriffspunkt. Ihre Lambda-Funktion erhält diese URL als Eingabe, wenn die Funktion aufgerufen wird.

Sie können Ihre Lambda-Funktion so einstellen, dass diese vorsignierte URL zum Abrufen des ursprünglichen Objekts verwendet wird, anstatt S3 direkt aufzurufen. Mit diesem Modell können Sie bessere Sicherheitsgrenzen auf Ihre Objekte anwenden. Sie können den direkten Objektzugriff über S3 Buckets oder S3-Zugriffspunkte auf einen begrenzten Satz von IAM-Rollen oder Benutzern beschränken. Dieser Ansatz schützt Ihre Lambda-Funktionen auch davor, dem Problem des verwirrten Stellvertreters ausgesetzt zu sein, bei dem eine falsch konfigurierte Funktion mit anderen

Berechtigungen als Ihr Aufrufer den Zugriff auf Objekte zulassen oder verweigern könnte, wenn dies nicht der Fall ist.

Öffentlicher Zugriff auf Objekt-Lambda-Zugriffspunkt

S3 Object Lambda erlaubt keinen anonymen oder öffentlichen Zugriff, da Amazon S3 Ihre Identität autorisieren muss, um eine S3-Object-Lambda-Anforderung abzuschließen. Wenn Sie Anforderungen über einen Object Lambda Access Point aufrufen, benötigen Sie die Berechtigung `lambda:InvokeFunction` für die konfigurierte Lambda-Funktion. Entsprechend müssen Sie beim Aufrufen anderer API-Operationen über einen Object Lambda Access Point über die erforderlichen `s3:*`-Berechtigungen verfügen.

Ohne diese Berechtigungen schlagen Anforderungen zum Aufrufen von Lambda oder Delegieren an S3 als „HTTP 403 Verboten“-Fehler fehl. Jeder Zugriff muss von authentifizierten Prinzipalen erfolgen. Wenn Sie einen öffentlichen Zugang benötigen, können Sie Lambda@Edge als mögliche Alternative verwenden. Weitere Informationen finden Sie unter Anpassen am Edge mit Lambda@Edge im Amazon- CloudFront Entwicklerhandbuch.

IP-Adressen von Object Lambda Access Points

Die `describe-managed-prefix-lists`-Subnetze unterstützen Gateway-Endpunkte der Virtual Private Cloud (VPC) und beziehen sich auf die Routingtabelle der VPC-Endpunkte. Da der Object-Lambda-Zugangspunkt keine Gateway-VPC unterstützt, fehlen dessen IP-Bereiche. Die fehlenden Bereiche gehören zu Amazon S3, werden jedoch von Gateway-VPC-Endpunkten nicht unterstützt. Weitere Informationen zu finden Sie `describe-managed-prefix-lists`unter DescribeManagedPrefixLists in der Amazon EC2-API-Referenz und unter AWS IP-Adressbereiche im Allgemeine AWS-Referenz.

Konfigurieren von IAM-Richtlinien für Object Lambda Access Points

Amazon S3-Zugriffspunkte unterstützen AWS Identity and Access Management (IAM)-Ressourcenrichtlinien, mit denen Sie die Verwendung des Zugriffspunkts nach Ressource, Benutzer oder anderen Bedingungen steuern können. Sie können den Zugriff über eine optionale Ressourcenrichtlinie auf Ihrem Object Lambda Access Point oder eine Ressourcenrichtlinie für unterstützende Zugriffspunkte steuern. step-by-step Beispiele finden Sie unter Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda und Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend.

Die vier folgenden Ressourcen müssen über Berechtigungen verfügen, um mit Object Lambda Access Points zu arbeiten:

- Die IAM-Identität, z. B. Benutzer oder Rolle. Weitere Informationen zu unterschiedlichen IAM-Identitäten und bewährten Methoden finden Sie unter IAM-Identitäten (Benutzer, Benutzergruppen und Rollen) im IAM-Benutzerhandbuch.
- Der Bucket und der zugehörige Standardzugriffspunkt Wenn Sie mit Object Lambda Access Point arbeiten, wird dieser Standardzugriffspunkt als unterstützender Zugriffspunkt bezeichnet.
- Der Object Lambda Access Point.
- Die AWS Lambda Funktion.

> **Important**
>
> Bevor Sie Ihre Richtlinie speichern, stellen Sie sicher, dass Sie Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge von beheben AWS Identity and Access Management Access Analyzer. IAM Access Analyzer führt Richtlinienprüfungen durch, um Ihre Richtlinie anhand der IAM-Richtliniengrammatik und der bewährten Methoden zu validieren. Diese Prüfungen generieren Ergebnisse und bieten umsetzbare Empfehlungen, die Sie beim Erstellen von Richtlinien unterstützen, die funktionsfähig sind und den bewährten Methoden für Sicherheit entsprechen.
> Weitere Informationen zum Validieren von Richtlinien mit IAM Access Analyzer finden Sie unter Validierung der IAM-Access-Analyzer-Richtlinien im IAM-Benutzerhandbuch. Eine Liste der Warnungen, Fehler und Vorschläge, die von IAM Access Analyzer zurückgegeben werden, finden Sie unter IAM-Access-Analyzer-Richtlinienprüfungsreferenz.

In den nachstehenden Richtlinienbeispielen wird davon ausgegangen, dass Sie die folgenden Ressourcen haben:

- Ein Amazon-S3-Bucket mit folgendem Amazon-Ressourcennamen (ARN):

  `arn:aws:s3:::DOC-EXAMPLE-BUCKET1`
- Ein Amazon-S3-Standard-Zugriffspunkt für diesen Bucket mit dem folgenden ARN:

  `arn:aws:s3:us-east-1:111122223333:accesspoint/my-access-point`
- Object Lambda Access Point mit dem folgenden ARN:

  `arn:aws:s3-object-lambda:us-east-1:111122223333:accesspoint/my-object-lambda-ap`

- Eine - AWS Lambda Funktion mit dem folgenden ARN:

  `arn:aws:lambda:us-east-1:111122223333:function:MyObjectLambdaFunction`

> **Note**
>
> Wenn Sie eine Lambda-Funktion aus Ihrem Konto verwenden, müssen Sie die Funktionsversion in Ihre Richtlinien-Anweisung einfügen. Im folgenden Beispiel-ARN ist die Version durch $LATEST angegeben:
> `arn:aws:lambda:us-east-1:111122223333:function:MyObjectLambdaFunction:$LATEST`
> Weitere Informationen zu Lambda-Funktionsversionen finden Sie unter Lambda-Funktionsversionen im AWS Lambda -Entwicklerhandbuch.

Example – Bucket-Richtlinie zum Delegieren der Zugriffskontrolle an Standardzugriffspunkte

Das folgende Beispiel für eine S3-Bucket-Richtlinie delegiert die Zugriffskontrolle für einen Bucket an die Standardzugriffspunkte des Buckets. Diese Richtlinie ermöglicht Vollzugriff auf alle Zugriffspunkte, die dem Konto des Bucket-Eigentümers gehören. Somit wird der gesamte Zugriff auf diesen Bucket durch die an seine Zugriffspunkte angehängten Richtlinien gesteuert. Benutzer können nur über einen Zugriffspunkt aus dem Bucket lesen, was bedeutet, dass Operationen nur über Zugriffspunkte aufgerufen werden können. Weitere Informationen finden Sie unter Delegieren der Zugangskontrolle an Zugriffspunkte.

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS":"account-ARN"},
        "Action" : "*",
        "Resource" : [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
        ],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointAccount" : "Bucket owner's account
 ID" }
        }
```

```
    }]
}
```

Example – IAM-Richtlinie, die einem Benutzer die erforderlichen Berechtigungen zur Verwendung eines Object Lambda Access Point gewährt

Die folgende IAM-Richtlinie erteilt einem Benutzer Berechtigungen für die Lambda-Funktion, den Standardzugriffspunkt und den Object Lambda Access Point.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowLambdaInvocation",
      "Action": [
        "lambda:InvokeFunction"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:lambda:us-
east-1:111122223333:function:MyObjectLambdaFunction:$LATEST",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": [
            "s3-object-lambda.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "AllowStandardAccessPointAccess",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:us-east-1:111122223333:accesspoint/my-access-point/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": [
            "s3-object-lambda.amazonaws.com"
          ]
        }
      }
```

```
    },
    {
      "Sid": "AllowObjectLambdaAccess",
      "Action": [
        "s3-object-lambda:Get*",
        "s3-object-lambda:List*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3-object-lambda:us-east-1:111122223333:accesspoint/my-
object-lambda-ap"
    }
  ]
}
```

Aktivieren von Berechtigungen für Lambda-Ausführungsrollen

Wenn `GET` Anforderungen an einen Object Lambda Access Point gestellt werden, benötigt Ihre Lambda-Funktion die Berechtigung zum Senden von Daten an den S3 Object Lambda Access Point. Diese Berechtigung wird durch Aktivieren der `s3-object-lambda:WriteGetObjectResponse`-Berechtigung für die Ausführungsrolle Ihrer Lambda-Funktion bereitgestellt. Sie können eine neue Ausführungsrolle erstellen oder eine vorhandene Rolle aktualisieren.

> **Note**
>
> Ihre Funktion benötigt die `s3-object-lambda:WriteGetObjectResponse`-Berechtigung nur, wenn Sie eine `GET`-Anforderung stellen.

So erstellen Sie eine Ausführungsrolle in der IAM-Konsole

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Wählen Sie Rolle erstellen aus.
4. Wählen Sie unter Häufige Anwendungsfälle die Option Lambda aus.
5. Wählen Sie Weiter aus.
6. Suchen Sie auf der Seite Berechtigungen hinzufügen nach der AWS `AmazonS3ObjectLambdaExecutionRolePolicy`verwalteten Richtlinie und aktivieren Sie dann das Kontrollkästchen neben dem Richtliniennamen.

Diese Richtlinie sollte die Aktion `s3-object-lambda:WriteGetObjectResponse` enthalten.

7. Wählen Sie Weiter aus.
8. Geben Sie auf der Seite Name, review, and create (Name, prüfen und erstellen) für Role name (Rollenname) **`s3-object-lambda-role`** ein.
9. (Optional) Fügen Sie eine Beschreibung und Tags für diese Rolle hinzu.
10. Wählen Sie Rolle erstellen aus.
11. Wenden Sie die neu erstellte **`s3-object-lambda-role`** als Ausführungsrolle Ihrer Lambda-Funktion. Dies kann während oder nach der Erstellung der Lambda-Funktion in der Lambda-Konsole erfolgen.

Weitere Informationen zu Ausführungsrollen finden Sie unter Lambda-Ausführungsrolle im Entwicklerhandbuch für AWS Lambda .

Verwenden von Kontextschlüsseln mit Object Lambda Access Points

S3 Object Lambda wertet Kontextschlüssel wie `s3-object-lambda:TlsVersion` oder `s3-object-lambda:AuthType` im Zusammenhang mit der Verbindung oder dem Signieren der Anfrage aus. Alle anderen Kontextschlüssel, wie z. B. `s3:prefix`, werden von Amazon S3 ausgewertet.

Unterstützung von CORS in Object Lambda Access Point

Wenn S3 Object Lambda eine Anforderung von einem Browser empfängt oder die Anforderung einen `Origin`-Header enthält, fügt S3 Object Lambda immer das Header-Feld `“AllowedOrigins":"*"` hinzu.

Weitere Informationen finden Sie unter Cross-Origin Resource Sharing (CORS) verwenden.

Schreiben von Lambda-Funktionen für S3 Object Lambda Access Points

In diesem Abschnitt wird beschrieben, wie Sie AWS Lambda Funktionen zur Verwendung mit Amazon S3 Object Lambda Access Points schreiben.

Weitere Informationen zu vollständigen end-to-end Verfahren für einige S3-Objekt-Lambda-Aufgaben finden Sie im Folgenden:

- Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda

- Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend
- Tutorial: Verwenden von S3 Object Lambda, um Bilder beim Abrufen dynamisch mit Wasserzeichen zu versehen

Themen

- Arbeiten mit GetObject-Anforderungen in Lambda
- Arbeiten mit HeadObject-Anforderungen in Lambda
- Arbeiten mit ListObjects-Anforderungen in Lambda
- Arbeiten mit ListObjectsV2-Anforderungen in Lambda
- Format und Verwendung des Ereigniskontexts
- Arbeiten mit Range- und partNumber-Headern

Arbeiten mit **`GetObject`**-Anforderungen in Lambda

In diesem Abschnitt wird davon ausgegangen, dass Ihr Object Lambda Access Point für den Aufruf der Lambda-Funktion für `GetObject` konfiguriert ist. S3 Objekt Lambda enthält die Amazon-S3-API-Operation `WriteGetObjectResponse`, mit der die Lambda-Funktion dem `GetObject`-Aufrufer benutzerdefinierte Daten und Antwort-Header bereitstellen kann.

`WriteGetObjectResponse` bietet Ihnen eine umfassende Kontrolle über den Statuscode, die Antwort-Header und den Antworttext, basierend auf Ihren Verarbeitungsanforderungen. Sie können `WriteGetObjectResponse` verwenden, um mit dem gesamten transformierten Objekt, Teilen des transformierten Objekts oder anderen Antworten basierend auf dem Kontext Ihrer Anwendung zu antworten. Der folgende Abschnitt zeigt eindeutige Beispiele für die Verwendung des `WriteGetObjectResponse`-API-Vorgangs.

- Beispiel 1: Mit einem HTTP-Statuscode 403 (Forbidden) antworten
- Beispiel 2: Reagieren Sie mit einem transformierten Bild
- Beispiel 3: Komprimierte Inhalte streamen

Beispiel 1: Mit einem HTTP-Statuscode 403 (Forbidden) antworten

Sie können `WriteGetObjectResponse` verwenden, um basierend auf dem Inhalt des Objekts mit dem HTTP-Statuscode 403 (Forbidden) zu antworten.

Java

```
package com.amazon.s3.objectlambda;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.events.S3ObjectLambdaEvent;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.WriteGetObjectResponseRequest;

import java.io.ByteArrayInputStream;
import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;

public class Example1 {

    public void handleRequest(S3ObjectLambdaEvent event, Context context) throws
 Exception {
        AmazonS3 s3Client = AmazonS3Client.builder().build();

        // Check to see if the request contains all of the necessary information.
        // If it does not, send a 4XX response and a custom error code and message.
        // Otherwise, retrieve the object from S3 and stream it
        // to the client unchanged.
        var tokenIsNotPresent = !
event.getUserRequest().getHeaders().containsKey("requiredToken");
        if (tokenIsNotPresent) {
            s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                    .withRequestRoute(event.outputRoute())
                    .withRequestToken(event.outputToken())
                    .withStatusCode(403)
                    .withContentLength(0L).withInputStream(new
 ByteArrayInputStream(new byte[0]))
                    .withErrorCode("MissingRequiredToken")
                    .withErrorMessage("The required token was not present in the
 request."));
            return;
        }

        // Prepare the presigned URL for use and make the request to S3.
        HttpClient httpClient = HttpClient.newBuilder().build();
```

```
        var presignedResponse = httpClient.send(
                HttpRequest.newBuilder(new URI(event.inputS3Url())).GET().build(),
                HttpResponse.BodyHandlers.ofInputStream());

        // Stream the original bytes back to the caller.
        s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                .withRequestRoute(event.outputRoute())
                .withRequestToken(event.outputToken())
                .withInputStream(presignedResponse.body()));
    }
}
```

Python

```
import boto3
import requests

def handler(event, context):
    s3 = boto3.client('s3')

    """
    Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    should be delivered and contains a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    The 'userRequest' object has information related to the user who made this
 'GetObject' request to
    S3 Object Lambda.
    """
    get_context = event["getObjectContext"]
    user_request_headers = event["userRequest"]["headers"]

    route = get_context["outputRoute"]
    token = get_context["outputToken"]
    s3_url = get_context["inputS3Url"]

    # Check for the presence of a 'CustomHeader' header and deny or allow based on
 that header.
    is_token_present = "SuperSecretToken" in user_request_headers

    if is_token_present:
        # If the user presented our custom 'SuperSecretToken' header, we send the
 requested object back to the user.
```

```
        response = requests.get(s3_url)
        s3.write_get_object_response(RequestRoute=route, RequestToken=token,
Body=response.content)
    else:
        # If the token is not present, we send an error back to the user.
        s3.write_get_object_response(RequestRoute=route, RequestToken=token,
StatusCode=403,
        ErrorCode="NoSuperSecretTokenFound", ErrorMessage="The request was not
secret enough.")

    # Gracefully exit the Lambda function.
    return { 'status_code': 200 }
```

Node.js

```
const { S3 } = require('aws-sdk');
const axios = require('axios').default;

exports.handler = async (event) => {
    const s3 = new S3();

    // Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    // should be delivered and contains a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    // The 'userRequest' object has information related to the user who made this
'GetObject' request to S3 Object Lambda.
    const { userRequest, getObjectContext } = event;
    const { outputRoute, outputToken, inputS3Url } = getObjectContext;

    // Check for the presence of a 'CustomHeader' header and deny or allow based on
 that header.
    const isTokenPresent = Object
        .keys(userRequest.headers)
        .includes("SuperSecretToken");

    if (!isTokenPresent) {
        // If the token is not present, we send an error back to the user. The
'await' in front of the request
        // indicates that we want to wait for this request to finish sending before
 moving on.
        await s3.writeGetObjectResponse({
            RequestRoute: outputRoute,
```

```
            RequestToken: outputToken,
            StatusCode: 403,
            ErrorCode: "NoSuperSecretTokenFound",
            ErrorMessage: "The request was not secret enough.",
        }).promise();
    } else {
        // If the user presented our custom 'SuperSecretToken' header, we send the
 requested object back to the user.
        // Again, note the presence of 'await'.
        const presignedResponse = await axios.get(inputS3Url);
        await s3.writeGetObjectResponse({
            RequestRoute: outputRoute,
            RequestToken: outputToken,
            Body: presignedResponse.data,
        }).promise();
    }

    // Gracefully exit the Lambda function.
    return { statusCode: 200 };
}
```

Beispiel 2: Reagieren Sie mit einem transformierten Bild

Wenn Sie eine Bildtransformation durchführen, stellen Sie möglicherweise fest, dass Sie alle Bytes des Quellobjekts benötigen, bevor Sie mit der Verarbeitung beginnen können. In diesem Fall gibt Ihre `WriteGetObjectResponse`-Anforderung das gesamte Objekt in einem Aufruf an die anfragende Anwendung zurück.

Java

```
package com.amazon.s3.objectlambda;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.events.S3ObjectLambdaEvent;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.WriteGetObjectResponseRequest;

import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
import java.awt.Image;
```

```
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;

public class Example2 {

    private static final int HEIGHT = 250;
    private static final int WIDTH = 250;

    public void handleRequest(S3ObjectLambdaEvent event, Context context) throws
 Exception {
        AmazonS3 s3Client = AmazonS3Client.builder().build();
        HttpClient httpClient = HttpClient.newBuilder().build();

        // Prepare the presigned URL for use and make the request to S3.
        var presignedResponse = httpClient.send(
                HttpRequest.newBuilder(new URI(event.inputS3Url())).GET().build(),
                HttpResponse.BodyHandlers.ofInputStream());

        // The entire image is loaded into memory here so that we can resize it.
        // Once the resizing is completed, we write the bytes into the body
        // of the WriteGetObjectResponse request.
        var originalImage = ImageIO.read(presignedResponse.body());
        var resizingImage = originalImage.getScaledInstance(WIDTH, HEIGHT,
 Image.SCALE_DEFAULT);
        var resizedImage = new BufferedImage(WIDTH, HEIGHT,
 BufferedImage.TYPE_INT_RGB);
        resizedImage.createGraphics().drawImage(resizingImage, 0, 0, WIDTH, HEIGHT,
 null);

        var baos = new ByteArrayOutputStream();
        ImageIO.write(resizedImage, "png", baos);

        // Stream the bytes back to the caller.
        s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                .withRequestRoute(event.outputRoute())
                .withRequestToken(event.outputToken())
                .withInputStream(new ByteArrayInputStream(baos.toByteArray())));
    }
}
```

Python

```
import boto3
import requests
import io
from PIL import Image

def handler(event, context):
    """
    Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    should be delivered and has a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    The 'userRequest' object has information related to the user who made this
 'GetObject' request to
    S3 Object Lambda.
    """
    get_context = event["getObjectContext"]
    route = get_context["outputRoute"]
    token = get_context["outputToken"]
    s3_url = get_context["inputS3Url"]

    """
    In this case, we're resizing .png images that are stored in S3 and are
 accessible through the presigned URL
    'inputS3Url'.
    """
    image_request = requests.get(s3_url)
    image = Image.open(io.BytesIO(image_request.content))
    image.thumbnail((256,256), Image.ANTIALIAS)

    transformed = io.BytesIO()
    image.save(transformed, "png")

    # Send the resized image back to the client.
    s3 = boto3.client('s3')
    s3.write_get_object_response(Body=transformed.getvalue(), RequestRoute=route,
 RequestToken=token)

    # Gracefully exit the Lambda function.
    return { 'status_code': 200 }
```

Node.js

```
const { S3 } = require('aws-sdk');
const axios = require('axios').default;
const sharp = require('sharp');

exports.handler = async (event) => {
    const s3 = new S3();

    // Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    // should be delivered and has a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    const { getObjectContext } = event;
    const { outputRoute, outputToken, inputS3Url } = getObjectContext;

    // In this case, we're resizing .png images that are stored in S3 and are
 accessible through the presigned URL
    // 'inputS3Url'.
    const { data } = await axios.get(inputS3Url, { responseType: 'arraybuffer' });

    // Resize the image.
    const resized = await sharp(data)
        .resize({ width: 256, height: 256 })
        .toBuffer();

    // Send the resized image back to the client.
    await s3.writeGetObjectResponse({
        RequestRoute: outputRoute,
        RequestToken: outputToken,
        Body: resized,
    }).promise();

    // Gracefully exit the Lambda function.
    return { statusCode: 200 };
}
```

Beispiel 3: Komprimierte Inhalte streamen

Beim Komprimieren von Objekten werden komprimierte Daten inkrementell erzeugt. Folglich können Sie Ihre `WriteGetObjectResponse`-Anforderung verwenden, um die komprimierten Daten

zurückzugeben, sobald sie bereit sind. Wie in diesem Beispiel gezeigt, ist es nicht notwendig, die Länge der abgeschlossenen Transformation zu kennen.

Java

```
package com.amazon.s3.objectlambda;

import com.amazonaws.services.lambda.runtime.events.S3ObjectLambdaEvent;
import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.WriteGetObjectResponseRequest;

import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;

public class Example3 {

    public void handleRequest(S3ObjectLambdaEvent event, Context context) throws
 Exception {
        AmazonS3 s3Client = AmazonS3Client.builder().build();
        HttpClient httpClient = HttpClient.newBuilder().build();

        // Request the original object from S3.
        var presignedResponse = httpClient.send(
                HttpRequest.newBuilder(new URI(event.inputS3Url())).GET().build(),
                HttpResponse.BodyHandlers.ofInputStream());

        // Consume the incoming response body from the presigned request,
        // apply our transformation on that data, and emit the transformed bytes
        // into the body of the WriteGetObjectResponse request as soon as they're
 ready.
        // This example compresses the data from S3, but any processing pertinent
        // to your application can be performed here.
        var bodyStream = new GZIPCompressingInputStream(presignedResponse.body());

        // Stream the bytes back to the caller.
        s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                .withRequestRoute(event.outputRoute())
                .withRequestToken(event.outputToken())
                .withInputStream(bodyStream));
```

```
    }

}
```

Python

```
import boto3
import requests
import zlib
from botocore.config import Config


"""
A helper class to work with content iterators. Takes an interator and compresses the
 bytes that come from it. It
implements 'read' and '__iter__' so that the SDK can stream the response.
"""
class Compress:
    def __init__(self, content_iter):
        self.content = content_iter
        self.compressed_obj = zlib.compressobj()

    def read(self, _size):
        for data in self.__iter__()
            return data

    def __iter__(self):
        while True:
            data = next(self.content)
            chunk = self.compressed_obj.compress(data)
            if not chunk:
                break

            yield chunk

        yield self.compressed_obj.flush()


def handler(event, context):
    """
    Setting the 'payload_signing_enabled' property to False allows us to send a
 streamed response back to the client.
```

```
   in this scenario, a streamed response means that the bytes are not buffered into
memory as we're compressing them,
   but instead are sent straight to the user.
   """
   my_config = Config(
       region_name='eu-west-1',
       signature_version='s3v4',
       s3={
           "payload_signing_enabled": False
       }
   )
   s3 = boto3.client('s3', config=my_config)

   """
   Retrieve the operation context object from the event. This object indicates
where the WriteGetObjectResponse request
   should be delivered and has a presigned URL in 'inputS3Url' where we can
download the requested object from.
   The 'userRequest' object has information related to the user who made this
'GetObject' request to S3 Object Lambda.
   """
   get_context = event["getObjectContext"]
   route = get_context["outputRoute"]
   token = get_context["outputToken"]
   s3_url = get_context["inputS3Url"]

   # Compress the 'get' request stream.
   with requests.get(s3_url, stream=True) as r:
       compressed = Compress(r.iter_content())

       # Send the stream back to the client.
       s3.write_get_object_response(Body=compressed, RequestRoute=route,
RequestToken=token, ContentType="text/plain",
                                    ContentEncoding="gzip")

   # Gracefully exit the Lambda function.
   return {'status_code': 200}
```

Node.js

```
const { S3 } = require('aws-sdk');
const axios = require('axios').default;
const zlib = require('zlib');
```

```
exports.handler = async (event) => {
    const s3 = new S3();

    // Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    // should be delivered and has a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    const { getObjectContext } = event;
    const { outputRoute, outputToken, inputS3Url } = getObjectContext;

    // Download the object from S3 and process it as a stream, because it might be a
 huge object and we don't want to
    // buffer it in memory. Note the use of 'await' because we want to wait for
 'writeGetObjectResponse' to finish
    // before we can exit the Lambda function.
    await axios({
        method: 'GET',
        url: inputS3Url,
        responseType: 'stream',
    }).then(
        // Gzip the stream.
        response => response.data.pipe(zlib.createGzip())
    ).then(
        // Finally send the gzip-ed stream back to the client.
        stream => s3.writeGetObjectResponse({
            RequestRoute: outputRoute,
            RequestToken: outputToken,
            Body: stream,
            ContentType: "text/plain",
            ContentEncoding: "gzip",
        }).promise()
    );

    // Gracefully exit the Lambda function.
    return { statusCode: 200 };
}
```

Note

Obwohl S3 Object Lambda ermöglicht, bis zu 60 Sekunden eine vollständige Antwort an den Aufrufer über die `WriteGetObjectResponse`-Anforderung zu senden, ist die tatsächlich

verfügbare Zeit möglicherweise geringer. Zum Beispiel kann Ihr Timeout für Lambda-Funktionen weniger als 60 Sekunden betragen. In anderen Fällen kann der Aufrufer strengere Zeitbeschränkungen haben.

Damit der ursprüngliche Aufrufer eine andere Antwort als einen HTTP-Statuscode 500 (Interner Serverfehler) erhält, muss der `WriteGetObjectResponse`-Aufruf abgeschlossen sein. Wenn die Lambda-Funktion mit einer Ausnahme oder anderweitig Ergebnisse zurückgibt, bevor die API-Operation `WriteGetObjectResponse` aufgerufen wird, erhält der ursprüngliche Aufrufer eine 500-Antwort (Interner Serverfehler). Ausnahmen, die während der Zeit ausgelöst werden, die zum Abschließen der Antwort benötigt wird, führen zu verkürzten Antworten an den Aufrufer. Wenn die Lambda-Funktion eine HTTP-Statuscode 200 (OK)-Antwort vom `WriteGetObjectResponse`-API-Aufruf erhält, hat der ursprüngliche Aufrufer die vollständige Anforderung gesendet. Die Antwort der Lambda-Funktion, wird von S3 Object Lambda, unabhängig davon, ob eine Ausnahme ausgelöst wird oder nicht, ignoriert.

Beim Aufruf der API-Operation `WriteGetObjectResponse` benötigt Amazon S3 den Routen- und Anforderungstoken aus dem Ereigniskontext. Weitere Informationen finden Sie unter Format und Verwendung des Ereigniskontexts.

Die Routen- und Anforderungstokenparameter sind erforderlich, um die `WriteGetObjectResult`-Antwort mit dem ursprünglichen Aufrufer zu verbinden. Obwohl es immer angemessen ist, 500-Antworten (Interner Serverfehler) zu wiederholen, beachten Sie, dass es sich bei dem Anforderungstoken um ein einmaliges Verwendungstoken handelt und nachfolgende Verwendungsversuche zu HTTP-Statuscode 400 (Bad Request)-Antworten führen können. Obwohl der Aufruf an `WriteGetObjectResponse` mit den Routen- und Anforderungstoken nicht von der aufgerufenen Lambda-Funktion erfolgen muss, ist hierfür eine Identität im selben Konto erforderlich. Der Aufruf muss außerdem abgeschlossen sein, bevor die Lambda-Funktion die Ausführung beendet.

Arbeiten mit **`HeadObject`**-Anforderungen in Lambda

In diesem Abschnitt wird davon ausgegangen, dass Ihr Object Lambda Access Point für den Aufruf der Lambda-Funktion für `HeadObject` konfiguriert ist. Lambda erhält eine JSON-Nutzlast, die einen Schlüssel namens `headObjectContext` enthält. Innerhalb des Kontexts gibt es eine einzelne Eigenschaft namens `inputS3Url`, bei der es sich um eine vorsignierte URL für den unterstützenden Zugriffspunkt für `HeadObject` handelt.

Die vorsignierte URL enthält die folgenden Eigenschaften, wenn sie angegeben sind:

- `versionId` (in den Abfrageparametern)
- `requestPayer` (im `x-amz-request-payer`-Header)
- `expectedBucketOwner` (im `x-amz-expected-bucket-owner`-Header)

Andere Eigenschaften werden nicht vorsigniert und daher nicht berücksichtigt. Nicht signierte Optionen, die als Header gesendet werden, können manuell zur Anforderung hinzugefügt werden, wenn die vorsignierte URL aufgerufen wird, die sich in den `userRequest`-Headern befindet. Serverseitige Verschlüsselungsoptionen werden für `HeadObject` nicht unterstützt.

Informationen zu den URI-Parametern der Anforderungssyntax finden Sie unter `HeadObject` in der API-Referenz für Amazon Simple Storage Service.

Das folgende Beispiel zeigt eine Lambda-JSON-Eingabe-Nutzlast für `HeadObject`.

```
{
  "xAmzRequestId": "requestId",
  "**headObjectContext**": {
    "**inputS3Url**": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/example?X-Amz-Security-Token=<snip>"
  },
  "configuration": {
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
  },
  "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
        "headers": {
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
   },
   "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
```

```
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
             "attributes": {
             "mfaAuthenticated": "false",
             "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
        },
        "sessionIssuer": {
             "type": "Role",
             "principalId": "principalId",
             "arn": "arn:aws:iam::111122223333:role/Admin",
             "accountId": "111122223333",
             "userName": "Admin"
             }
        }
    },
  "protocolVersion": "1.00"
}
```

Ihre Lambda-Funktion sollte ein JSON-Objekt zurückgeben, das die Header und Werte enthält, die für den `HeadObject`-Aufruf zurückgegeben werden.

Das folgende Beispiel zeigt die Struktur der JSON-Nachricht der Lambda-Antwort für `HeadObject`.

```
{
    "statusCode": <number>; // Required
    "errorCode": <string>;
    "errorMessage": <string>;
    "headers": {
        "Accept-Ranges": <string>,
        "x-amz-archive-status": <string>,
        "x-amz-server-side-encryption-bucket-key-enabled": <boolean>,
        "Cache-Control": <string>,
        "Content-Disposition": <string>,
        "Content-Encoding": <string>,
        "Content-Language": <string>,
        "Content-Length": <number>, // Required
        "Content-Type": <string>,
        "x-amz-delete-marker": <boolean>,
        "ETag": <string>,
        "Expires": <string>,
        "x-amz-expiration": <string>,
        "Last-Modified": <string>,
```

```
        "x-amz-missing-meta": <number>,
        "x-amz-object-lock-mode": <string>,
        "x-amz-object-lock-legal-hold": <string>,
        "x-amz-object-lock-retain-until-date": <string>,
        "x-amz-mp-parts-count": <number>,
        "x-amz-replication-status": <string>,
        "x-amz-request-charged": <string>,
        "x-amz-restore": <string>,
        "x-amz-server-side-encryption": <string>,
        "x-amz-server-side-encryption-customer-algorithm": <string>,
        "x-amz-server-side-encryption-aws-kms-key-id": <string>,
        "x-amz-server-side-encryption-customer-key-MD5": <string>,
        "x-amz-storage-class": <string>,
        "x-amz-tagging-count": <number>,
        "x-amz-version-id": <string>,
        <x-amz-meta-headers>: <string>, // user-defined metadata
        "x-amz-meta-meta1": <string>, // example of the user-defined metadata header,
 it will need the x-amz-meta prefix
        "x-amz-meta-meta2": <string>
        ...
    };
}
```

Das folgende Beispiel zeigt, wie Sie die vorsignierte URL verwenden, um Ihre Antwort aufzufüllen, indem Sie die Header-Werte nach Bedarf ändern, bevor das JSON-Objekt zurückgegeben wird.

Python

```
import requests

def lambda_handler(event, context):
    print(event)

    # Extract the presigned URL from the input.
    s3_url = event["headObjectContext"]["inputS3Url"]

    # Get the head of the object from S3.
    response = requests.head(s3_url)

    # Return the error to S3 Object Lambda (if applicable).
    if (response.status_code >= 400):
        return {
            "statusCode": response.status_code,
```

```
            "errorCode": "RequestFailure",
            "errorMessage": "Request to S3 failed"
    }

    # Store the headers in a dictionary.
    response_headers = dict(response.headers)

    # This obscures Content-Type in a transformation, it is optional to add
    response_headers["Content-Type"] = ""

    # Return the headers to S3 Object Lambda.
    return {
        "statusCode": response.status_code,
        "headers": response_headers
        }
```

Arbeiten mit **`ListObjects`**-Anforderungen in Lambda

In diesem Abschnitt wird davon ausgegangen, dass Ihr Object Lambda Access Point für den Aufruf der Lambda-Funktion für `ListObjects` konfiguriert ist. Lambda erhält die JSON-Nutzlast mit einem neuen Objekt namens `listObjectsContext`. `listObjectsContext` enthält eine einzelne Eigenschaft, `inputS3Url`, bei der es sich um eine vorsignierte URL für den unterstützenden Zugriffspunkt `ListObjects` handelt.

Im Gegensatz zu `GetObject` und `HeadObject` enthält die vorsignierte URL die folgenden Eigenschaften, wenn sie angegeben sind:

- Alle Abfrageparameter
- `requestPayer` (im `x-amz-request-payer`-Header)
- `expectedBucketOwner` (im `x-amz-expected-bucket-owner`-Header)

Informationen zu den URI-Parametern der Anforderungssyntax finden Sie unter `ListObjects` in der API-Referenz für Amazon Simple Storage Service.

> **Important**
>
> Wir empfehlen, bei der Entwicklung von Anwendungen die neuere Version, ListObjectsV2, zu verwenden. Aus Gründen der Abwärtskompatibilität unterstützt Amazon S3 weiterhin `ListObjects`.

Das folgende Beispiel zeigt die Lambda-JSON-Eingabe-Nutzlast für `ListObjects`.

```
{
    "xAmzRequestId": "requestId",
     "**listObjectsContext**": {
     "**inputS3Url**": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/?X-Amz-Security-Token=<snip>",
     },
    "configuration": {
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
    },
    "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
        "headers": {
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
    },
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
```

```
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "principalId",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            }
        }
    },
  "protocolVersion": "1.00"
}
```

Ihre Lambda-Funktion sollte ein JSON-Objekt zurückgeben, das den Statuscode, das Listen-XML-Ergebnis oder die Fehlerinformationen enthält, die von S3 Object Lambda zurückgegeben werden.

S3 Object Lambda verarbeitet oder validiert `listResultXml` nicht, sondern leitet es an den `ListObjects`-Aufrufer weiter. Für `listBucketResult` erwartet S3 Object Lambda, dass bestimmte Eigenschaften von einem speziellen Typ sind, und löst Ausnahmen aus, wenn es sie nicht analysieren kann. `listResultXml` und `listBucketResult` können nicht gleichzeitig bereitgestellt werden.

Das folgende Beispiel zeigt, wie die vorsignierte URL verwendet wird, um Amazon S3 aufzurufen und das Ergebnis zum Auffüllen einer Antwort einschließlich der Fehlerprüfung zu nutzen.

Python

```
import requests
import xmltodict

def lambda_handler(event, context):
    # Extract the presigned URL from the input.
    s3_url = event["listObjectsContext"]["inputS3Url"]


    # Get the head of the object from Amazon S3.
    response = requests.get(s3_url)

    # Return the error to S3 Object Lambda (if applicable).
    if (response.status_code >= 400):
        error = xmltodict.parse(response.content)
        return {
```

```
            "statusCode": response.status_code,
            "errorCode": error["Error"]["Code"],
            "errorMessage": error["Error"]["Message"]
        }

    # Store the XML result in a dict.
    response_dict = xmltodict.parse(response.content)

    # This obscures StorageClass in a transformation, it is optional to add
    for item in response_dict['ListBucketResult']['Contents']:
        item['StorageClass'] = ""

    # Convert back to XML.
    listResultXml = xmltodict.unparse(response_dict)

    # Create response with listResultXml.
    response_with_list_result_xml = {
        'statusCode': 200,
        'listResultXml': listResultXml
    }

    # Create response with listBucketResult.
    response_dict['ListBucketResult'] =
 sanitize_response_dict(response_dict['ListBucketResult'])
    response_with_list_bucket_result = {
        'statusCode': 200,
        'listBucketResult': response_dict['ListBucketResult']
    }

    # Return the list to S3 Object Lambda.
    # Can return response_with_list_result_xml or response_with_list_bucket_result
    return response_with_list_result_xml

# Converting the response_dict's key to correct casing
def sanitize_response_dict(response_dict: dict):
    new_response_dict = dict()
    for key, value in response_dict.items():
        new_key = key[0].lower() + key[1:] if key != "ID" else 'id'
        if type(value) == list:
            newlist = []
            for element in value:
                if type(element) == type(dict()):
                    element = sanitize_response_dict(element)
                newlist.append(element)
```

```
            value = newlist
        elif type(value) == dict:
            value = sanitize_response_dict(value)
        new_response_dict[new_key] = value
    return new_response_dict
```

Das folgende Beispiel zeigt die Struktur der JSON-Nachricht der Lambda-Antwort für `ListObjects`.

```
{
  "statusCode": <number>; // Required
  "errorCode": <string>;
  "errorMessage": <string>;
  "listResultXml": <string>; // This can also be Error XML string in case S3 returned
 error response when calling the pre-signed URL

  "listBucketResult": {  // listBucketResult can be provided instead of listResultXml,
 however they can not both be provided in the JSON response
       "name": <string>,  // Required for 'listBucketResult'
       "prefix": <string>,
       "marker": <string>,
       "nextMarker": <string>,
       "maxKeys": <int>,   // Required for 'listBucketResult'
       "delimiter": <string>,
       "encodingType": <string>
       "isTruncated": <boolean>,  // Required for 'listBucketResult'
       "contents": [  {
           "key": <string>,  // Required for 'content'
           "lastModified": <string>,
           "eTag": <string>,
           "checksumAlgorithm": <string>,   // CRC32,  CRC32C,  SHA1,  SHA256
           "size": <int>,   // Required for 'content'
           "owner": {
               "displayName": <string>,  // Required for 'owner'
               "id": <string>,  // Required for 'owner'
           },
           "storageClass": <string>
           },
       ...
       ],
       "commonPrefixes": [  {
           "prefix": <string>   // Required for 'commonPrefix'
       },
```

```
        ...
        ],
    }
}
```

Arbeiten mit **`ListObjectsV2`**-Anforderungen in Lambda

In diesem Abschnitt wird davon ausgegangen, dass Ihr Object Lambda Access Point für den Aufruf der Lambda-Funktion für `ListObjectsV2` konfiguriert ist. Lambda erhält die JSON-Nutzlast mit einem neuen Objekt namens `listObjectsV2Context`. `listObjectsV2Context` enthält eine einzelne Eigenschaft, `inputS3Url`, bei der es sich um eine vorsignierte URL für den unterstützenden Zugriffspunkt `ListObjectsV2` handelt.

Im Gegensatz zu `GetObject` und `HeadObject` enthält die vorsignierte URL die folgenden Eigenschaften, wenn sie angegeben sind:

- Alle Abfrageparameter
- `requestPayer` (im `x-amz-request-payer`-Header)
- `expectedBucketOwner` (im `x-amz-expected-bucket-owner`-Header)

Informationen zu den URI-Parametern der Anforderungssyntax finden Sie unter `ListObjectsV2` in der API-Referenz für Amazon Simple Storage Service.

Das folgende Beispiel zeigt die Lambda-JSON-Eingabe-Nutzlast für `ListObjectsV2`.

```
{
    "xAmzRequestId": "requestId",
     "**listObjectsV2Context**": {
     "**inputS3Url**": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/?list-type=2&X-Amz-Security-Token=<snip>",
     },
    "configuration": {
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
    },
    "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
```

```
        "headers": {
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
    },
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "principalId",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            }
        }
    },
  "protocolVersion": "1.00"
}
```

Ihre Lambda-Funktion sollte ein JSON-Objekt zurückgeben, das den Statuscode, das Listen-XML-Ergebnis oder die Fehlerinformationen enthält, die von S3 Object Lambda zurückgegeben werden.

S3 Object Lambda verarbeitet oder validiert `listResultXml` nicht, sondern leitet es an den `ListObjectsV2`-Aufrufer weiter. Für `listBucketResult` erwartet S3 Object Lambda, dass bestimmte Eigenschaften von einem speziellen Typ sind, und löst Ausnahmen aus, wenn es sie nicht analysieren kann. `listResultXml` und `listBucketResult` können nicht gleichzeitig bereitgestellt werden.

Das folgende Beispiel zeigt, wie die vorsignierte URL verwendet wird, um Amazon S3 aufzurufen und das Ergebnis zum Auffüllen einer Antwort einschließlich der Fehlerprüfung zu nutzen.

Python

```
import requests
import xmltodict

def lambda_handler(event, context):
    # Extract the presigned URL from the input.
    s3_url = event["listObjectsV2Context"]["inputS3Url"]


    # Get the head of the object from Amazon S3.
    response = requests.get(s3_url)

    # Return the error to S3 Object Lambda (if applicable).
    if (response.status_code >= 400):
        error = xmltodict.parse(response.content)
        return {
            "statusCode": response.status_code,
            "errorCode": error["Error"]["Code"],
            "errorMessage": error["Error"]["Message"]
        }

    # Store the XML result in a dict.
    response_dict = xmltodict.parse(response.content)

    # This obscures StorageClass in a transformation, it is optional to add
    for item in response_dict['ListBucketResult']['Contents']:
        item['StorageClass'] = ""

    # Convert back to XML.
    listResultXml = xmltodict.unparse(response_dict)

    # Create response with listResultXml.
    response_with_list_result_xml = {
        'statusCode': 200,
        'listResultXml': listResultXml
    }

    # Create response with listBucketResult.
    response_dict['ListBucketResult'] =
 sanitize_response_dict(response_dict['ListBucketResult'])
    response_with_list_bucket_result = {
        'statusCode': 200,
        'listBucketResult': response_dict['ListBucketResult']
```

```
    }

    # Return the list to S3 Object Lambda.
    # Can return response_with_list_result_xml or response_with_list_bucket_result
    return response_with_list_result_xml

# Converting the response_dict's key to correct casing
def sanitize_response_dict(response_dict: dict):
    new_response_dict = dict()
    for key, value in response_dict.items():
        new_key = key[0].lower() + key[1:] if key != "ID" else 'id'
        if type(value) == list:
            newlist = []
            for element in value:
                if type(element) == type(dict()):
                    element = sanitize_response_dict(element)
                newlist.append(element)
            value = newlist
        elif type(value) == dict:
            value = sanitize_response_dict(value)
        new_response_dict[new_key] = value
    return new_response_dict
```

Das folgende Beispiel zeigt die Struktur der JSON-Nachricht der Lambda-Antwort für `ListObjectsV2`.

```
{
    "statusCode": <number>; // Required
    "errorCode": <string>;
    "errorMessage": <string>;
    "listResultXml": <string>; // This can also be Error XML string in case S3 returned
 error response when calling the pre-signed URL

    "listBucketResult": {  // listBucketResult can be provided instead of
 listResultXml, however they can not both be provided in the JSON response
        "name": <string>, // Required for 'listBucketResult'
        "prefix": <string>,
        "startAfter": <string>,
        "continuationToken": <string>,
        "nextContinuationToken": <string>,
        "keyCount": <int>, // Required for 'listBucketResult'
        "maxKeys": <int>, // Required for 'listBucketResult'
```

```
        "delimiter": <string>,
        "encodingType": <string>
        "isTruncated": <boolean>, // Required for 'listBucketResult'
        "contents": [ {
            "key": <string>, // Required for 'content'
            "lastModified": <string>,
            "eTag": <string>,
            "checksumAlgorithm": <string>, // CRC32, CRC32C, SHA1, SHA256
            "size": <int>, // Required for 'content'
            "owner": {
                "displayName": <string>, // Required for 'owner'
                "id": <string>, // Required for 'owner'
            },
            "storageClass": <string>
            },
            ...
        ],
        "commonPrefixes": [ {
            "prefix": <string> // Required for 'commonPrefix'
            },
        ...
        ],
    }
}
```

Format und Verwendung des Ereigniskontexts

Amazon S3 Object Lambda bietet Kontext über die Anforderung, die für den Fall gestellt wird, dass an Ihre AWS Lambda Funktion übergeben wird. Nachstehend finden Sie eine Beispielanforderung: Beschreibungen der Felder folgen nach dem Beispiel.

```
{
    "xAmzRequestId": "requestId",
    "getObjectContext": {
        "inputS3Url": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/example?X-Amz-Security-Token=<snip>",
        "outputRoute": "io-use1-001",
        "outputToken": "OutputToken"
    },
    "configuration": {
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
```

```
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
    },
    "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
        "headers": {
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
    },
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "principalId",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            }
        }
    },
    "protocolVersion": "1.00"
}
```


Die folgenden Felder sind in der Anforderung enthalten:

- `xAmzRequestId` – Die Amazon-S3-Anforderungs-ID für diese Anforderung. Wir empfehlen, diesen Wert zu protokollieren, um beim Debuggen zu helfen.
- `getObjectContext` – Die Eingabe- und Ausgabedetails für Verbindungen zu Amazon S3 und S3 Object Lambda.

  - `inputS3Url` – Eine vorsignierte URL, die verwendet werden kann, um das ursprüngliche Objekt von Amazon S3 abzurufen. Die URL wird mit der Identität des ursprünglichen Aufrufers signiert und die Berechtigungen des entsprechenden Benutzers gelten, wenn die URL verwendet wird. Wenn die URL signierte Header enthält, muss die Lambda-Funktion diese Header in den Aufruf von Amazon S3 aufnehmen, mit Ausnahme des `Host`-Headers.
  - `outputRoute` – Ein Routing-Token, das der Lambda-URL des S3-Objekts hinzugefügt wird, wenn die Lambda-Funktion `WriteGetObjectResponse` aufruft.
  - `outputToken` – Ein undurchsichtiges Token, das von S3 Object Lambda verwendet wird, um den `WriteGetObjectResponse`-Aufruf mit dem ursprünglichen Aufrufer abzugleichen.
- `configuration` – Informationen zur Konfiguration des Object Lambda Access Point.
  - `accessPointArn` – Der Amazon-Ressourcenname (ARN) des Object Lambda Access Point, der diese Anforderung erhalten hat.
  - `supportingAccessPointArn` – Der ARN des unterstützenden Zugriffspunkts, der in der Konfiguration des Object Lambda Access Point angegeben ist.
  - `payload` – Benutzerdefinierte Daten, die auf die Konfiguration des Object Lambda Access Point angewendet werden. S3 Object Lambda behandelt diese Daten als eine undurchsichtige Zeichenfolge, daher muss sie möglicherweise vor der Verwendung dekodiert werden.
- `userRequest` – Informationen über den ursprünglichen Aufruf von S3 Object Lambda.
  - `url` – Die dekodierte URL der Anforderung, wie sie von S3 Object Lambda empfangen wurde, ohne autorisierungsbezogene Abfrageparameter.
  - `headers` – Eine Zuordnung von Zeichenfolgen zu Zeichenfolgen, die die HTTP-Header und ihre Werte aus dem ursprünglichen Aufruf enthalten, ohne autorisierungsbezogene Header. Wenn derselbe Header mehrfach erscheint, werden die Werte von jeder Instance desselben Header zu einer durch Kommata getrennten Liste zusammengefasst. Der Fall der ursprünglichen Header wird in dieser Zuordnung beibehalten.
- `userIdentity` – Details zur die Identität, die den Aufruf von S3 Object Lambda getätigt hat. Weitere Informationen finden Sie unter Protokollieren von Datenereignissen für Trails im AWS CloudTrail -Benutzerhandbuch.
  - `type` – Die Art der Identität.
  - `accountId` – Die , AWS-Konto zu der die Identität gehört.
  - `userName` – Der Anzeigename der Identität, von der der Aufruf stammt.
  - `principalId` – Die eindeutige ID für die Entität, von der der Aufruf stammt.

  - `arn` – Der ARN des Prinzipals, von dem der Aufruf stammt. Der letzte Abschnitt des ARN enthält den Benutzer oder die Rolle, von dem/der der Aufruf stammt.
  - `sessionContext` – Erfolgte die Abfrage mittels temporärer Sicherheitsanmeldeinformationen, stellt dieses Element Informationen über die Sitzung bereit, das für diese Anmeldeinformationen erstellt wurde.
  - `invokedBy` – Der Name des AWS-Service , der die Anforderung gestellt hat, z. B. Amazon EC2 Auto Scaling oder AWS Elastic Beanstalk.
  - `sessionIssuer` – Erfolgte die Abfrage mittels temporärer Sicherheitsanmeldeinformationen, gibt dieses Element Auskunft darüber, wie die Anmeldeinformationen erhalten wurden.
- `protocolVersion` – Die Versions-ID des bereitgestellten Kontextes. Das Format des Felds ist `{Major Version}.{Minor Version}`. Die Nebenversionsnummern sind immer zweistellige Zahlen. Jede Entfernung oder Änderung der Semantik eines Feldes erfordert einen Sprung der Hauptversion und erfordert ein aktives Opt-In. Amazon S3 kann jederzeit neue Felder hinzufügen. Zu diesem Zeitpunkt kann es zu einem Sprung der Nebenversion kommen. Aufgrund der Art der Software-Einführung sehen Sie möglicherweise die Verwendung mehrerer Nebenversionen gleichzeitig.

Arbeiten mit Range- und partNumber-Headern

Wenn Sie mit großen Objekten in Amazon S3 Object Lambda arbeiten, können Sie den `Range`-HTTP-Header verwenden, um einen bestimmten Bytebereich von einem Objekt herunterzuladen. Sie können gleichzeitige Verbindungen zu Amazon S3 verwenden, um verschiedene Bytebereiche aus demselben Objekt abzurufen. Sie können auch den `partNumber`-Parameter (eine Ganzzahl zwischen 1 und 10 000) verwenden, der eine „ranged“ Anforderung für das angegebene Teil des Objekts ausführt.

Weil es mehrere Möglichkeiten gibt, eine Anforderung zu bearbeiten, die den `Range`- oder `partNumber`-Parameter beinhaltet, wendet S3 Object Lambda diese Parameter nicht auf das transformierte Objekt an. Stattdessen muss Ihre AWS Lambda Funktion diese Funktionalität nach Bedarf für Ihre Anwendung implementieren.

Wenn Sie die Parameter `Range` und `partNumber` mit S3 Object Lambda verwenden möchten, gehen Sie wie folgt vor:

- Aktivieren Sie diese Parameter in der Konfiguration Ihres Object Lambda Access Point.

- Schreiben Sie eine Lambda-Funktion, die Anforderungen verarbeiten kann, die diese Parameter enthalten.

In den folgenden Schritten wird beschrieben, wie Sie dies erreichen.

Schritt 1: Konfigurieren Ihres Object Lambda Access Point

Standardmäßig antworten Object Lambda Access Points mit dem HTTP-Statuscodefehler 501 (Nicht implementiert) auf jede `GetObject`- oder `HeadObject`-Anforderung, die entweder in den Headern oder in den Abfrageparametern einen `Range`- oder `partNumber`-Parameter enthält.

Damit ein Object-Lambda-Zugriffspunkt solche Anforderungen annehmen kann, müssen Sie `GetObject-Range`, `GetObject-PartNumber`, `HeadObject-Range` oder `HeadObject-PartNumber` in den `AllowedFeatures`-Abschnitt die Konfiguration Ihres Object-Lambda-Zugangspunkts einschließen. Weitere Informationen zur Aktualisierung der Konfiguration Ihres Object Lambda Access Point finden Sie unter Erstellen von Objekt-Lambda-Zugriffspunkten.

Schritt 2: Implementieren Sie das **`Range`** oder **`partNumber`** in Ihrer Lambda-Funktion

Wenn Ihr Object Lambda Access Point Ihre Lambda-Funktion mit einer `GetObject`- oder `HeadObject`-Bereichsanforderung aufruft, ist der `Range`- oder `partNumber`-Parameter im Ereigniskontext enthalten. Die Position des Parameters im Ereigniskontext hängt davon ab, welcher Parameter verwendet wurde und wie er in die ursprüngliche Anforderung an den Object Lambda Access Point aufgenommen wurde, wie in der folgenden Tabelle erläutert.

Parameter	Ort des Ereigniskontexts
`Range` (Header)	`userRequest.headers.Range`
`Range` (Abfragep arameter)	`userRequest.url` (Abfrageparameter `Range`)
`partNumber`	`userRequest.url` (Abfrageparameter `partNumber`)

> **Important**
>
> Die bereitgestellte vorsignierte URL für Ihren Object Lambda Access Point enthält nicht den `Range`- oder `partNumber`-Parameter aus der ursprünglichen Anforderung. In den folgenden Optionen erfahren Sie, wie Sie diese Parameter in Ihrer Funktion handhaben können.

Nachdem Sie den `Range`- oder `partNumber`-Wert extrahiert haben, können Sie einen der folgenden Ansätze wählen, basierend auf den Anforderungen Ihrer Anwendung:

A. Ordnen Sie den angeforderten **`Range`**- oder **`partNumber`**-Wert dem transformierten Objekt zu (empfohlen).

Die zuverlässigste Art, `Range`- oder `partNumber`-Anforderungen zu bearbeiten, ist folgende:

- Rufen Sie das vollständige Objekt aus Amazon S3 ab.
- Transformieren Sie das Objekt.
- Wenden Sie die angeforderten `Range`- oder `partNumber`-Parameter auf das transformierte Objekt an.

Verwenden Sie dazu die bereitgestellte vorsignierte URL, um das gesamte Objekt von Amazon S3 abzurufen und das Objekt dann nach Bedarf zu verarbeiten. Ein Beispiel für eine Lambda-Funktion, die einen `Range` Parameter auf diese Weise verarbeitet, finden Sie in diesem Beispiel im AWS Samples GitHub -Repository.

B. Ordnen Sie den angeforderten **`Range`** der vorsignierten URL zu.

In einigen Fällen kann Ihre Lambda-Funktion den angeforderten `Range` direkt der vorsignierten URL zuordnen, um nur einen Teil des Objekts von Amazon S3 abzurufen. Dieser Ansatz ist nur dann geeignet, wenn Ihre Transformation die beiden folgenden Kriterien erfüllt:

1. Ihre Transformationsfunktion kann auf partielle Objektbereiche angewendet werden.
2. Das Anwenden der `Range`-Parameter vor oder nach der Transformationsfunktion führt zu demselben transformierten Objekt.

Beispielsweise erfüllt eine Transformationsfunktion, die alle Zeichen in einem ASCII-kodierten Objekt in Großbuchstaben konvertiert, beide vorhergehenden Kriterien. Die Transformation kann auf einen Teil eines Objekts angewendet werden, und durch Anwenden des `Range`-Parameters vor der Transformation wird das gleiche Ergebnis erzielt wie beim Anwenden des Parameters nach der Transformation.

Im Gegensatz dazu erfüllt eine Funktion, die die Zeichen in einem ASCII-kodierten Objekt umkehrt, diese Kriterien nicht. Eine solche Funktion erfüllt Kriterium 1, da sie auf partielle Objektbereiche angewendet werden kann. Sie erfüllt jedoch nicht Kriterium 2, da das Anwenden des `Range`-Parameters vor der Transformation andere Ergebnisse erzielt als das Anwenden des Parameters nach der Transformation.

Betrachten Sie eine Anforderung, die Funktion auf die ersten drei Zeichen eines Objekts mit dem Inhalt `abcdefg` anzuwenden. Das Anwenden des `Range`-Parameters vor der Transformation ruft nur `abcab` und kehrt dann die Daten um und gibt `cba` zurück. Wenn der Parameter jedoch nach der Transformation angewendet wird, ruft die Funktion das gesamte Objekt ab, kehrt es um und wendet dann den `Range`-Parameter, um `gfe` zurückzugeben. Da diese Ergebnisse unterschiedlich sind, sollte diese Funktion den `Range`-Parameter beim Abrufen des Objekts von Amazon S3 nicht anwenden. Stattdessen sollte es das gesamte Objekt abrufen, die Transformation durchführen und erst dann den `Range`-Parameter anwenden.

> **Warning**
>
> In vielen Fällen führt die Anwendung des `Range`-Parameters für die vorsignierte URL zu einem unerwarteten Verhalten durch die Lambda-Funktion oder den anfordernden Client. Wenn Sie nicht sicher sind, dass Ihre Anwendung ordnungsgemäß funktioniert, wenn Sie nur ein Teilobjekt von Amazon S3 abrufen, empfehlen wir Ihnen, vollständige Objekte abzurufen und zu transformieren, wie zuvor in Ansatz A beschrieben.

Wenn Ihre Anwendung die zuvor in Ansatz B beschriebenen Kriterien erfüllt, können Sie Ihre AWS Lambda Funktion vereinfachen, indem Sie nur den angeforderten Objektbereich abrufen und dann Ihre Transformation in diesem Bereich ausführen.

Im folgenden Java-Codebeispiel wird folgende Vorgehensweise gezeigt:

- Rufen Sie den `Range`-Header aus der `GetObject`-Anforderung ab.
- Fügen Sie der vorsignierten URL den `Range`-Header hinzu, mit dem Lambda den angeforderten Bereich von Amazon S3 abrufen kann.

```
private HttpRequest.Builder applyRangeHeader(ObjectLambdaEvent event,
 HttpRequest.Builder presignedRequest) {
    var header = event.getUserRequest().getHeaders().entrySet().stream()
            .filter(e -> e.getKey().toLowerCase(Locale.ROOT).equals("range"))
```

```
            .findFirst();

    // Add check in the query string itself.
    header.ifPresent(entry -> presignedRequest.header(entry.getKey(),
 entry.getValue()));
    return presignedRequest;
}
```

Verwenden von AWS erstellten Lambda-Funktionen

AWS bietet einige vorgefertigte AWS Lambda Funktionen, die Sie mit Amazon S3 Object Lambda verwenden können, um persönlich identifizierbare Informationen (PII) zu erkennen und zu redigieren und S3-Objekte zu dekomprimieren. Diese Lambda-Funktionen sind in AWS Serverless Application Repository verfügbar. Sie können diese Funktionen über die AWS Management Console auswählen, wenn Sie Ihren Object Lambda Access Point erstellen.

Weitere Informationen zum Bereitstellen von Serverless-Anwendungen über die finden Sie AWS Serverless Application Repository unter Bereitstellen von Anwendungen im AWS Serverless Application Repository -Entwicklerhandbuch.

> **Note**
>
> Die folgenden Beispiele können nur mit `GetObject`-Anforderungen verwendet werden.

Beispiel 1: PII-Zugriffskontrolle

Die Lambda-Funktion nutzt Amazon Comprehend, ein Natural Language Processing (NLP)-Service, der Machine Learning nutzt, um Einsichten und Zusammenhänge im Text zu finden. Diese Funktion erkennt automatisch persönlich identifizierbare Informationen (PII) wie Namen, Adressen, Daten, Kreditkartennummern und Sozialversicherungsnummern aus Dokumenten in Ihrem Amazon-S3-Bucket. Wenn Sie Dokumente in Ihrem Bucket haben, die PII enthalten, können Sie die PII-Zugriffskontrollfunktion konfigurieren, um diese PII-Entitätstypen zu erkennen und den Zugriff auf nicht autorisierte Benutzer zu beschränken.

Stellen Sie zunächst die folgende Lambda-Funktion in Ihrem Konto bereit und fügen Sie der Konfiguration Ihres Object Lambda Access Point den Amazon-Ressourcennamen (ARN) für die Funktion hinzu.

Es folgt ein Beispiel für einen ARN für diese Funktion:

```
arn:aws:serverlessrepo:us-east-1:111122223333:applications/
ComprehendPiiAccessControlS3ObjectLambda
```

Sie können diese Funktion auf dem hinzufügen oder anzeigen, AWS Management Console indem Sie den folgenden AWS Serverless Application Repository Link verwenden: ComprehendPiiAccessControlS3ObjectLambda.

Informationen zum Anzeigen dieser Funktion auf GitHubfinden Sie unter Amazon Comprehend S3 Object Lambda.

Beispiel 2: PII-Redigierung

Die Lambda-Funktion nutzt Amazon Comprehend, ein Natural Language Processing (NLP)-Service, der Machine Learning nutzt, um Einsichten und Zusammenhänge im Text zu finden. Sie redigiert automatisch persönlich identifizierbare Informationen (PII) wie Namen, Adressen, Daten, Kreditkartennummern und Sozialversicherungsnummern aus Dokumenten in Ihrem Amazon-S3-Bucket.

Wenn Sie Dokumente in Ihrem Bucket haben, die Informationen wie Kreditkartennummern oder Bankkontoinformationen enthalten, können Sie die PII-Redigierungs-S3 Object Lambda-Funktion konfigurieren, um PII zu erkennen, und dann eine Kopie dieser Dokumente zurückgeben, in denen die Typen von PII-Entitäten redigiert werden.

Stellen Sie zunächst die folgende Lambda-Funktion in Ihrem Konto bereit und fügen Sie der Konfiguration Ihres Object Lambda Access Point den ARN für die Fuktion hinzu.

Es folgt ein Beispiel für einen ARN für diese Funktion:

```
arn:aws:serverlessrepo:us-east-1:111122223333::applications/
ComprehendPiiRedactionS3ObjectLambda
```

Sie können diese Funktion über den folgenden AWS Serverless Application Repository Link in AWS Management Console der hinzufügen oder anzeigen: ComprehendPiiRedactionS3ObjectLambda.

Informationen zum Anzeigen dieser Funktion auf GitHubfinden Sie unter Amazon Comprehend S3 Object Lambda.

Weitere Informationen zu vollständigen end-to-end Verfahren für einige S3-Objekt-Lambda-Aufgaben bei der PII-Schwärzung finden Sie unter Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend.

Beispiel 3: Dekompression

Die Lambda-Funktion `S3ObjectLambdaDecompression` kann Objekte, die in Amazon S3 gespeichert sind, in einem von sechs komprimierten Dateiformaten dekomprimieren: `bzip2`, `gzip`, `snappy`, `zlib`, `zstandard` und `ZIP`.

Stellen Sie zunächst die folgende Lambda-Funktion in Ihrem Konto bereit und fügen Sie der Konfiguration Ihres Object Lambda Access Point den ARN für die Fuktion hinzu.

Es folgt ein Beispiel für einen ARN für diese Funktion:

```
arn:aws:serverlessrepo:us-east-1:111122223333::applications/S3ObjectLambdaDecompression
```

Sie können diese Funktion auf dem hinzufügen oder anzeigen, AWS Management Console indem Sie den folgenden AWS Serverless Application Repository Link verwenden: S3ObjectLambdaDecompression.

Informationen zum Anzeigen dieser Funktion auf GitHubfinden Sie unter S3 Object Lambda Decompression.

Bewährte Methoden und Richtlinien für S3 Object Lambda

Befolgen Sie bei der Verwendung von S3 Object Lambda diesen bewährten Methoden und Richtlinien, um den Betrieb und die Leistung zu optimieren.

Themen

- Arbeiten mit S3 Object Lambda
- AWS-Services wird in Verbindung mit S3 Object Lambda verwendet
- Range- und partNumber-Header
- Transformieren von expiry-date
- Arbeiten mit den AWS CLI und AWS SDKs

Arbeiten mit S3 Object Lambda

S3 Object Lambda unterstützt nur die Verarbeitung von `GET`-, `LIST`- und `HEAD`-Anforderungen. Alle anderen Anforderungen rufen nicht auf AWS Lambda und geben stattdessen standardmäßige, nicht transformierte API-Antworten zurück. Sie können maximal 1 000 Object Lambda Access Points AWS-Konto pro und Region erstellen. Die AWS Lambda Funktion, die Sie verwenden, muss sich in demselben AWS-Konto und derselben Region wie der Object Lambda Access Point befinden.

S3 Object Lambda lässt bis zu 60 Sekunden zu, um eine vollständige Antwort zu seinen Aufrufer zu streamen. Ihre Funktion unterliegt auch AWS Lambda Standardkontingenten. Weitere Informationen finden Sie unter Lambda quotas (Lambda-Kontingente) im AWS Lambda -Entwicklerhandbuch.

Mit S3 Object Lambda wird Ihre angegebene Lambda-Funktion aufgerufen. Sie sind dafür verantwortlich, dass alle Daten, die von Ihrer angegebenen Lambda-Funktion oder -Anwendung aus S3 überschrieben oder gelöscht werden, beabsichtigt und korrekt sind.

Sie können S3 Object Lambda nur verwenden, um Operationen auf Objekten durchzuführen. Sie können S3 Object Lambda nicht verwenden, um andere Amazon-S3-Operationen auszuführen, z. B. das Ändern oder Löschen von Buckets. Eine vollständige Liste der S3-Vorgänge, die Zugriffspunkte unterstützen, finden Sie unter Zugriffspunkt-Kompatibilität mit S3-Vorgänge.

Zusätzlich zu dieser Liste unterstützen Object Lambda Access Points die API-Operationen `POST Object`, `CopyObject` (als Quelle) und `SelectObjectContent` nicht.

AWS-Services wird in Verbindung mit S3 Object Lambda verwendet

S3 Object Lambda verbindet Amazon S3 und optional andere AWS-Services Ihrer Wahl, um Objekte bereitzustellen AWS Lambda, die für die anfordernden Anwendungen relevant sind. Alle , die mit S3 Object Lambda AWS-Services verwendet werden, unterliegen ihren jeweiligen Service Level Agreements (SLAs). Wenn beispielsweise ein seine Service-Verpflichtung AWS-Service nicht erfüllt, können Sie eine Service-Gutschrift erhalten, wie in der SLA des Services dokumentiert.

`Range`- und **`partNumber`**-Header

Wenn Sie mit großen Objekten arbeiten, können Sie den `Range`-HTTP-Header verwenden, um einen bestimmten Bytebereich von einem Objekt herunterzuladen. Wenn Sie den `Range`-Header verwenden, ruft Ihre Anforderung nur den angegebenen Teil des Objekts ab. Mit dem `partNumber`-Header können Sie auch eine Bereichsanforderung für das angegebene Teil aus dem Objekt ausführen.

Weitere Informationen finden Sie unter Arbeiten mit Range- und partNumber-Headern.

Transformieren von **`expiry-date`**

Sie können transformierte Objekte von Ihrem Object Lambda Access Point auf der öffnen oder herunterladen AWS Management Console. Diese Objekte dürfen nicht abgelaufen sein. Wenn Ihre Lambda-Funktion das `expiry-date` Ihrer Objekte transformiert, sehen Sie möglicherweise abgelaufene Objekte, die nicht geöffnet oder heruntergeladen werden können. Dieses Verhalten gilt nur für wiederhergestellte Objekte von S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive.

Arbeiten mit den AWS CLI und AWS SDKs

AWS Command Line Interface (AWS CLI) S3-Unterbefehle (`cp`, `mv`und `sync`) und die Verwendung der AWS SDK for Java `TransferManager` Klasse werden für die Verwendung mit S3 Object Lambda nicht unterstützt.

Tutorials zu S3 Object Lambda

In den folgenden Tutorials werden vollständige end-to-end Verfahren für einige S3-Objekt-Lambda-Aufgaben vorgestellt.

- Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda
- Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend
- Tutorial: Verwenden von S3 Object Lambda, um Bilder beim Abrufen dynamisch mit Wasserzeichen zu versehen

Debuggen von S3 Object Lambda

Anforderungen an Zugriffspunkte von Amazon S3 Object Lambda können zu neuen Fehlerantworten führen, wenn mit dem Aufruf oder der Ausführung der Lambda-Funktion etwas schief geht. Diese Fehler folgen dem gleichen Format wie die Standardfehler von Amazon S3. Weitere Informationen zu Fehlern für S3 Object Lambda finden Sie unter Fehlercodeliste für S3 Object Lambda in der API-Referenz zum Amazon Simple Storage Service.

Weitere Informationen zum allgemeinen Debuggen von Lambda-Funktionen finden Sie unter Monitoring and troubleshooting Lambda applications (Überwachung und Fehlerbehebung bei Lambda-Anwendungen) im AWS Lambda -Entwicklerhandbuch.

Informationen zu standardmäßigen Amazon-S3-Fehlern finden Sie unter Fehlerantworten in der API-Referenz zum Amazon Simple Storage Service.

Sie können Anforderungsmetriken in Amazon CloudWatch für Ihre Object Lambda Access Points aktivieren. Diese Metriken helfen Ihnen dabei, die Betriebsleistung Ihrer Zugriffspunkte zu überwachen. Sie können Anforderungsmetriken während oder nach der Erstellung Ihres Object Lambda Access Point aktivieren. Weitere Informationen finden Sie unter Anforderungsmetriken für S3 Object Lambda in CloudWatch.

Um eine detailliertere Protokollierung der an Ihre Object Lambda Access Points gesendeten Anforderungen zu erhalten, können Sie AWS CloudTrail -Datenereignisse aktivieren. Weitere Informationen finden Sie unter Protokollieren von Datenereignissen für Trails im AWS CloudTrail -Benutzerhandbuch.

Informationen zu S3-Objekt-Lambda-Tutorials finden Sie im Folgenden:

- Tutorial: Transformieren von Daten für Ihre Anwendung mit S3 Object Lambda
- Tutorial: Erkennen und Redigieren von PII-Daten mit S3 Object Lambda und Amazon Comprehend
- Tutorial: Verwenden von S3 Object Lambda, um Bilder beim Abrufen dynamisch mit Wasserzeichen zu versehen

Weitere Hinweise zu standardmäßigen Zugriffspunkten finden Sie unter Verwalten des Datenzugriffs mit Amazon S3-Zugangspunkten.

Hinweise zum Arbeiten mit Buckets finden Sie unter Bucket-Übersicht. Weitere Informationen zur Arbeit mit Objekten finden Sie unter Übersicht über Amazon-S3-Objekte.

Was ist S3 Express One Zone?

Amazon S3 Express One Zone ist eine leistungsstarke Amazon-S3-Speicherklasse mit einer einzelnen Zone, die speziell für den konsistenten Datenzugriff im einstelligen Millisekundenbereich für Ihre latenzempfindlichsten Anwendungen entwickelt wurde. S3 Express One Zone ist die heute verfügbare Cloud-Objekt-Speicherklasse mit der niedrigsten Latenz, mit bis zu zehnmal schnelleren Datenzugriffsgeschwindigkeiten und mit Anforderungskosten, die 50 Prozent niedriger sind als S3 Standard. Anwendungen können sofort davon profitieren, dass Anfragen bis zu einer Größenordnung schneller abgeschlossen werden. S3 Express One Zone bietet ähnliche Leistungselastizität wie andere S3-Speicherklassen.

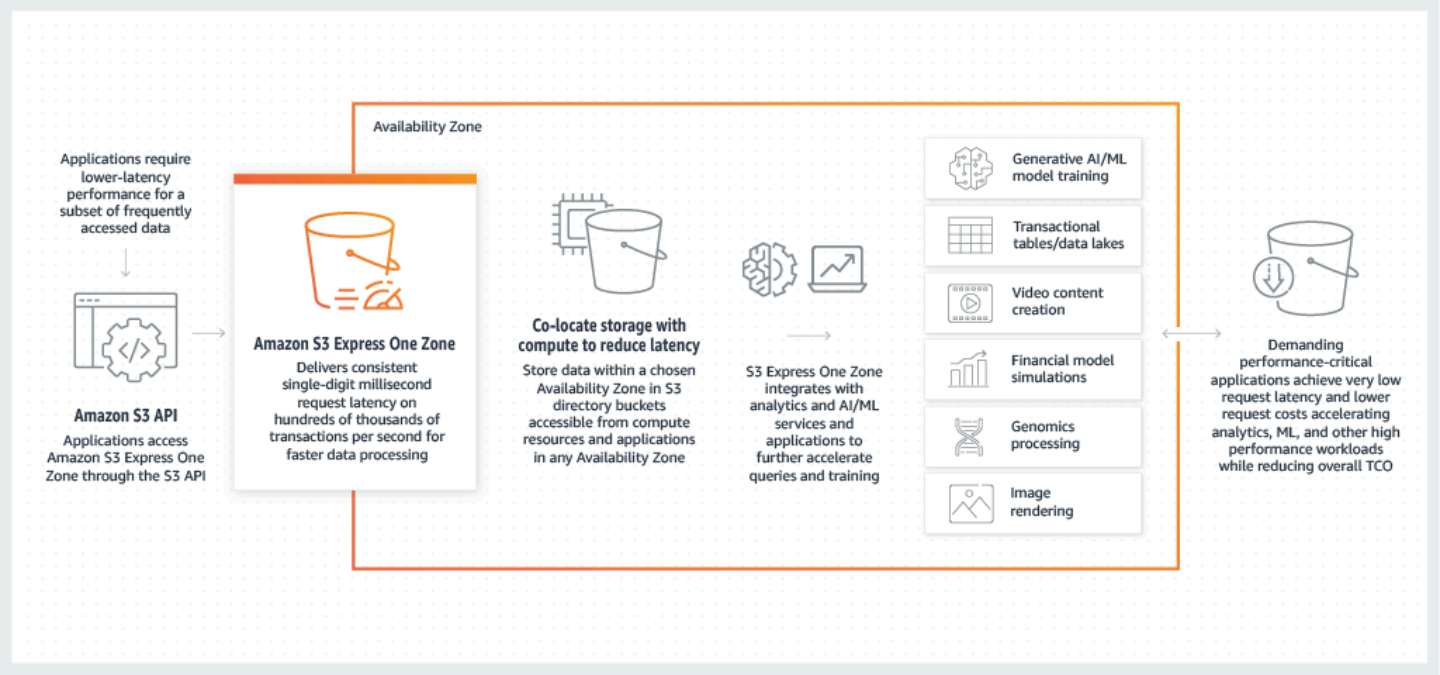
Wie bei anderen Amazon S3-Speicherklassen müssen Sie keine Kapazitäts- oder Durchsatzanforderungen im Voraus planen oder bereitstellen. Sie können Ihren Speicher je nach Bedarf hoch- oder herunterskalieren und über die Amazon S3-API auf Ihre Daten zugreifen.

S3 Express One Zone ist die erste S3-Speicherklasse, bei der Sie eine einzelne Availability Zone mit der Option auswählen können, Ihren Objektspeicher gemeinsam mit Ihren Computingressourcen zu platzieren, was die höchstmögliche Zugriffsgeschwindigkeit bietet. Um die Zugriffsgeschwindigkeit weiter zu erhöhen und Hunderttausende von Anfragen pro Sekunde zu unterstützen, werden Daten in der Speicherklasse S3 Express One Zone in einem neuen Bucket-Typ gespeichert: einem Amazon S3-Verzeichnis-Bucket. Jeder Verzeichnis-Bucket kann Hunderttausende von Transaktionen pro Sekunde (TPS) unterstützen, unabhängig von Schlüsselnamen oder Zugriffsmustern.

Die Speicherklasse Amazon S3 Express One Zone ist für eine Verfügbarkeit von 99,95 Prozent innerhalb einer einzigen Availability Zone konzipiert und wird durch das Amazon S3 Service Level Agreement unterstützt. Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. S3 Express One Zone wurde entwickelt, um mit gleichzeitigen Geräteausfällen umzugehen, indem verlorene Redundanz schnell erkannt und behoben werden kann. Wenn das vorhandene Gerät ausfällt, leitet S3 Express One Zone Anfragen automatisch an neue Geräte innerhalb einer Availability Zone weiter. Diese Redundanz trägt dazu bei, den unterbrechungsfreien Zugriff auf Ihre Daten innerhalb einer Availability Zone sicherzustellen.

S3 Express One Zone ist ideal für jede Anwendung geeignet, bei der es wichtig ist, die für den Zugriff auf ein Objekt erforderliche Latenz zu minimieren. Bei solchen Anwendungen kann es sich um Workflows handeln, bei denen Menschen interagieren, z. B. bei der Videobearbeitung, bei denen Entwickler einen reaktionsschnellen Zugriff auf Inhalte über ihre Benutzeroberflächen benötigen. S3 Express One Zone bietet auch Vorteile für Analysen und maschinelles Lernen, die

ähnliche Anforderungen an die Reaktionsfähigkeit ihrer Daten stellen, insbesondere Workloads mit vielen kleineren Zugriffen oder einer großen Anzahl zufälliger Zugriffe. S3 Express One Zone kann zusammen mit anderen verwendet werdenAWS-Services, um Analysen und Workloads für künstliche Intelligenz und Machine Learning (AI/ML) zu unterstützen, z. B. Amazon EMR, Amazon SageMakerund Amazon Athena.



Wenn Sie S3 Express One Zone verwenden, können Sie mit Ihrem Verzeichnis-Bucket in einer Virtual Private Cloud (VPC) interagieren, indem Sie einen Gateway-VPC-Endpunkt verwenden. Mit einem Gateway-Endpunkt können Sie von Ihrer VPC aus ohne Internet-Gateway oder NAT-Gerät für Ihre VPC und ohne zusätzliche Kosten auf S3 Express One Zone-Verzeichnis-Buckets zugreifen.

Sie können viele der gleichen Amazon S3-API-Operationen und -Funktionen mit Verzeichnis-Buckets verwenden, die Sie mit Allzweck-Buckets und anderen Speicherklassen verwenden. Dazu gehören Mountpoint für Amazon S3, serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3), S3 Batch Operations und S3 Block Public Access. Sie können auf S3 Express One Zone zugreifen, indem Sie die Amazon S3-Konsole, AWS Command Line Interface (AWS CLI), -AWSSDKs und die Amazon S3-REST-API verwenden. SDKs

Weitere Informationen zu S3 Express One Zone finden Sie in den folgenden Themen.

- Übersicht
- Funktionen von S3 Express One Zone
- Zugehörige Services

- Nächste Schritte

Übersicht

Um die Leistung zu optimieren und die Latenz zu reduzieren, führt S3 Express One Zone die folgenden neuen Konzepte ein.

Einzelne Availability Zone

Die Speicherklasse Amazon S3 Express One Zone ist für eine Verfügbarkeit von 99,95 Prozent innerhalb einer einzigen Availability Zone konzipiert und wird durch das Amazon S3 Service Level Agreement unterstützt. Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. S3 Express One Zone wurde entwickelt, um mit gleichzeitigen Geräteausfällen umzugehen, indem verlorene Redundanz schnell erkannt und behoben werden kann. Wenn das vorhandene Gerät ausfällt, leitet S3 Express One Zone Anfragen automatisch an neue Geräte innerhalb einer Availability Zone weiter. Diese Redundanz trägt dazu bei, den unterbrechungsfreien Zugriff auf Ihre Daten innerhalb einer Availability Zone sicherzustellen.

Eine Availability Zone ist eines oder mehrere diskrete Rechenzentren mit redundanter Stromversorgung, Vernetzung und Konnektivität in einem AWS-Region. Wenn Sie einen Verzeichnis-Bucket erstellen, wählen Sie die Availability Zone und die AWS-Region für Ihren Bucket.

Verzeichnis-Buckets

Es gibt zwei Arten von Amazon S3-Buckets: S3-Allzweck-Buckets und S3-Verzeichnis-Buckets. Allzweck-Buckets sind der standardmäßige Amazon-S3-Bucket-Typ, der für die überwiegende Mehrheit der S3-Anwendungsfälle verwendet wird. Verzeichnis-Buckets verwenden die Speicherklasse S3 Express One Zone, die für Workloads oder leistungskritische Anwendungen konzipiert ist, die eine konsistente Latenz im einstelligen Millisekundenbereich erfordern. Wählen Sie den Bucket-Typ aus, der Ihren Anwendungs- und Leistungsanforderungen am besten entspricht.

Verzeichnis-Buckets organisieren Daten hierarchisch in Verzeichnissen, im Gegensatz zur flachen Speicherstruktur von Allzweck-Buckets. Es gibt keine Präfixbeschränkungen für Verzeichnis-Buckets und einzelne Verzeichnisse können horizontal skaliert werden.

Verzeichnis-Buckets verwenden die Speicherklasse S3 Express One Zone, die für die Verwendung durch leistungssensitive Anwendungen konzipiert wurde. Mit S3 Express One Zone können Sie eine einzelne Availability Zone mit der Option auswählen, Ihren Objektspeicher gemeinsam mit

Ihren Rechenressourcen zu platzieren, was die höchstmögliche Zugriffsgeschwindigkeit bietet. Dies ist im Gegensatz zu Allzweck-Buckets, die Objekte redundant über mehrere Availability Zones in speichernAWS-Regionen.

Weitere Information zu Verzeichnis-Buckets finden Sie unter Verzeichnis-Buckets. Weitere Informationen zu Allzweck-Buckets finden Sie unter Bucket-Übersicht.

Endpunkte und Gateway-VPC-Endpunkte

API-Operationen zur Bucket-Verwaltung für Verzeichnis-Buckets sind über einen regionalen Endpunkt verfügbar und werden als API-Operationen für regionale Endpunkte bezeichnet. Beispiele für API-Operationen für regionalen Endpunkte sind `CreateBucket` und `DeleteBucket`. Nachdem Sie einen Verzeichnis-Bucket erstellt haben, können Sie mithilfe von API-Operationen für zonale Endpunkte die Objekte in Ihrem Verzeichnis-Bucket hochladen und verwalten. API-Vorgänge für zonale Endpunkte sind über einen zonalen Endpunkt verfügbar. Beispiele für API-Operationen für zonale Endpunkte sind `PutObject` und `CopyObject`.

Sie können von Ihrer VPC aus über Gateway-VPC-Endpunkte auf S3 Express One Zone zugreifen. Nachdem Sie den Gateway-Endpunkt erstellt haben, können Sie ihn als Ziel in Ihrer Routing-Tabelle für Datenverkehr hinzufügen, der von Ihrer VPC zu S3Express One Zone bestimmt ist. Wie bei Amazon S3 fallen für die Nutzung von Gateway-Endpunkten keine zusätzlichen Gebühren an. Weitere Informationen dazu, wie Sie Gateway-VPC-Endpunkte konfigurieren, finden Sie unter Networking für S3 Express One Zone.

Sitzungsbasierte Autorisierung

Mit S3 Express One Zone authentifizieren und autorisieren Sie Anfragen über einen neuen sitzungsbasierten Mechanismus, der für die geringste Latenz optimiert ist. Sie können mit `CreateSession` temporäre Anmeldeinformationen anfordern, die den Zugriff auf Ihren Bucket mit geringer Latenz ermöglichen. Diese temporären Anmeldeinformationen sind einem bestimmten S3-Verzeichnis-Bucket zugeordnet. Sitzungstoken werden nur mit zonalen Operationen (Objektebene) verwendet (mit Ausnahme von CopyObject). Weitere Informationen finden Sie unter `CreateSession`-Autorisierung.

Die unterstützten AWS SDKs für S3 Express One Zone übernehmen die Sitzungseinrichtung und -aktualisierung in Ihrem Namen. Um Ihre Sitzungen zu schützen, laufen temporäre Sicherheitsanmeldedaten nach 5 Minuten ab. Nachdem Sie die AWS SDKs heruntergeladen und installiert und die erforderlichen AWS Identity and Access Management (IAM)-Berechtigungen konfiguriert haben, können Sie sofort mit der Verwendung von API-Operationen beginnen.

Funktionen von S3 Express One Zone

Die folgenden S3-Funktionen sind für S3 Express One Zone verfügbar. Eine vollständige Liste der unterstützten API-Operationen und nicht unterstützten Funktionen finden Sie unter Wodurch zeichnet sich S3 Express One Zone aus?.

Zugriffsverwaltung und Sicherheit

Mit Verzeichnis-Buckets können Sie die folgenden Funktionen zum Überwachen und Verwalten des Zugriffs verwenden. Standardmäßig sind Verzeichnis-Buckets privat und sind nur für Benutzer zugänglich, denen explizit Zugriff gewährt wurde. Im Gegensatz zu Allzweck-Buckets, bei denen die Zugriffskontrollgrenze auf Bucket-, Präfix- oder Objekt-Tag-Ebene festgelegt werden kann, wird die Zugriffskontrollgrenze für Verzeichnis-Buckets nur auf Bucket-Ebene festgelegt. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

- S3 Block Public Access – Alle Einstellungen für S3 Block Public Access sind standardmäßig auf Bucket-Ebene aktiviert. Diese Standardeinstellung kann nicht geändert werden.
- S3 Object Ownership (standardmäßig vom Bucket-Eigentümer erzwungen) – Zugriffssteuerungslisten (ACLs) werden für Verzeichnis-Buckets nicht unterstützt. Verzeichnis-Buckets verwenden automatisch die Einstellung „Bucket-Eigentümer erzwungen“ für S3 Object Ownership. „Bucket-Eigentümer erzwungen“ bedeutet, dass ACLs deaktiviert sind und der Bucket-Eigentümer automatisch Eigentümer jedes Objekts im Bucket ist und die volle Kontrolle über dieses besitzt. Diese Standardeinstellung kann nicht geändert werden.
- AWS Identity and Access Management (IAM) – IAM hilft Ihnen, den Zugriff auf Ihre Verzeichnis-Buckets sicher zu kontrollieren. Sie können IAM verwenden, um über die `-s3express:CreateSession`Aktion Zugriff auf (regionale) Bucket-Management-API-Operationen und (zonale) Objektverwaltungs-API-Operationen zu gewähren. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.
  Im Gegensatz zu Objektverwaltungsaktionen können Bucket-Verwaltungsaktionen nicht kontoübergreifend sein. Nur der Bucket-Eigentümer kann diese Aktionen ausführen.
- Bucket-Richtlinien – Verwenden Sie die IAM-basierte Richtliniensprache, um ressourcenbasierte Berechtigungen für Ihre Verzeichnis-Buckets zu konfigurieren. Sie können IAM auch verwenden, um den Zugriff auf die `CreateSession` -API-Operation zu steuern, mit der Sie die zonalen API-Operationen oder die Objektverwaltung verwenden können. Sie können konto- oder kontoübergreifenden Zugriff auf zonale API-Operationen gewähren. Weitere Informationen zu

Berechtigungen und Richtlinien für S3 Express One Zone finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

- IAM Access Analyzer für S3 – Bewerten und überwachen Sie Ihre Zugriffsrichtlinien, um sicherzustellen, dass die Richtlinien nur den beabsichtigten Zugriff auf Ihre S3-Ressourcen bieten.

Protokollierung und Überwachung

S3 Express One Zone verwendet die folgenden S3-Protokollierungs- und Überwachungstools, mit denen Sie überwachen und steuern können, wie Ihre Ressourcen verwendet werden:

- Amazon- CloudWatch Metriken – Überwachen Sie Ihre AWS Ressourcen und Anwendungen, indem Sie verwenden CloudWatch , um Metriken zu erfassen und zu verfolgen. S3 Express One Zone verwendet denselben CloudWatch Namespace wie andere Amazon S3-Speicherklassen (`AWS/S3`) und unterstützt tägliche Speichermetriken für Verzeichnis-Buckets: `BucketSizeBytes` und `NumberOfObjects`. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.
- -AWS CloudTrailProtokolle – AWS CloudTrail ist ein AWS-Service , der Sie bei der Implementierung von Betriebs- und Risikoprüfungen, Governance und Compliance Ihres unterstützt, AWS-Konto indem er die von einem Benutzer, einer Rolle oder einem durchgeführten Aktionen aufzeichnetAWS-Service. Für S3 Express One Zone CloudTrail erfasst regionale Endpunkt-API-Operationen (z. B. `CreateBucket` und `PutBucketPolicy`) als Verwaltungsereignisse. Zu diesen Ereignissen gehören Aktionen, die in den AWS API-Operationen AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDKs und durchgeführt wurden. SDKs Die `eventsource` für CloudTrail Verwaltungsereignisse für S3 Express One Zone ist `s3express.amazonaws.com`. Weitere Informationen finden Sie unter Amazon S3 CloudTrail -Ereignisse.

> **Note**
>
> Amazon S3-Serverzugriffsprotokolle werden von S3 Express One Zone nicht unterstützt.

Verwaltung von Objekten

Nachdem Sie einen Verzeichnis-Bucket erstellt haben, können Sie Ihren Objektspeicher mithilfe der Amazon S3-Konsole, der AWS SDKs und verwaltenAWS CLI. SDKs Die folgenden Funktionen sind für die Objektverwaltung mit S3 Express One Zone verfügbar:

- S3 Batch Operations – Verwenden Sie Batch Operations, um Massenvorgänge für Objekte in Verzeichnis-Buckets durchzuführen, z. B. die Funktionen Kopieren und Aufrufen AWS Lambda. Sie können mit Batch Operations beispielsweise Objekte zwischen Verzeichnis-Buckets und Allzweck-Buckets kopieren. Mit Batchoperationen können Sie Milliarden von Objekten in großem Umfang mit einer einzigen S3-Anforderung verwalten, indem Sie die AWS -SDKs oder AWS CLI oder einige Klicks in der Amazon S3-Konsole verwenden.
- Import – Nachdem Sie einen Verzeichnis-Bucket erstellt haben, können Sie Ihren Bucket mithilfe der Importfunktion in der Amazon-S3-Konsole mit Objekten füllen. Import ist eine optimierte Methode zur Erstellung von Batch-Operations-Aufträgen zum Kopieren von Objekten aus Allzweck-Buckets in Verzeichnis-Buckets.

AWS-SDKs und Client-Bibliotheken

Nachdem Sie einen Verzeichnis-Bucket erstellt und ein Objekt in Ihren Bucket hochgeladen haben, können Sie Ihren Objektspeicher wie folgt verwalten.

- Mountpoint für Amazon S3 – Mountpoint für Amazon S3 ist ein Open-Source-Dateiclient, der Zugriff mit hohem Durchsatz ermöglicht und so die Rechenkosten für Data Lakes auf Amazon S3 senkt. Mountpoint für Amazon S3 übersetzt lokale Dateisystem-API-Aufrufe in S3-Objekt-API-Aufrufe wie `GET` und `LIST`. Es ist ideal für leseintensive Data-Lake-Workloads, die Petabyte an Daten verarbeiten und den hohen elastischen Durchsatz benötigen, der von Amazon S3 bereitgestellt wird, um über Tausende von Instances hoch- und herunterzuskalieren.
- S3A – S3A ist eine empfohlene Hadoop-kompatible Schnittstelle für den Zugriff auf Datenspeicher in Amazon S3. S3A ersetzt den S3N Hadoop Dateisystem-Client.
- PyTorch on AWS – PyTorch auf AWS ist ein Open-Source-Deep-Learning-Framework, das die Entwicklung von Machine-Learning-Modellen und deren Bereitstellung in der Produktion vereinfacht.
- AWS -SDKs – Sie können die AWS -SDKs bei der Entwicklung von Anwendungen mit Amazon S3 verwenden. Die AWS-SDKs vereinfachen Ihre Programmieraufgaben, indem sie die zugrunde

liegende Amazon-S3-REST-API umhüllen. Weitere Informationen zur Verwendung der AWS -SDKs mit S3 Express One Zone finden Sie unter the section called “AWS SDKs”.

Verschlüsselung und Datenschutz

In Verzeichnis-Buckets gespeicherte Objekte werden automatisch mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt. Verzeichnis-Buckets unterstützen keine serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) oder serverseitige Dual-Layer-Verschlüsselung mit AWS KMS keys (DSSE-KMS). Weitere Informationen finden Sie unter Datenschutz und Verschlüsselung und Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

S3 Express One Zone bietet Ihnen die Möglichkeit, den Prüfsummenalgorithmus auszuwählen, der zur Validierung Ihrer Daten beim Hoch- oder Herunterladen verwendet wird. Sie können einen der folgenden Secure Hash Algorithms (SHA)- oder Cyclic Redundancy Check (CRC)-Algorithmen zur Überprüfung der Datenintegrität auswählen: CRC32, CRC32C, SHA-1 und SHA-256. MD5-based Prüfsummen werden von der Speicherklasse S3 Express One Zone nicht unterstützt.

Weitere Informationen finden Sie unter Bewährte Methoden für zusätzliche S3-Prüfsummen.

AWS Signature Version 4 (SigV4)

S3 Express One Zone verwendet AWS Signature Version 4 (SigV4). SigV4 ist ein Signaturprotokoll, das zur Authentifizierung von Anforderungen an Amazon S3 über HTTPS verwendet wird. S3 Express One Zone signiert Anforderungen mithilfe von AWS Sigv4. Weitere Informationen finden Sie unter Authenticating Requests (Authentifizierung von Anforderungn) (AWS Signature Version 4) in der API-Referenz für Amazon Simple Storage Service.

Starke Konsistenz

S3 Express One Zone bietet eine starke read-after-write Konsistenz für - `PUT` und -`DELETE`Anforderungen von Objekten in Ihren Verzeichnis-Buckets in allen AWS-Regionen. Weitere Informationen finden Sie unter Amazon S3-Datenkonsistenzmodell.

Zugehörige Services

Sie können die folgenden AWS-Services mit der Speicherklasse S3 Express One Zone verwenden, um Ihren spezifischen Anwendungsfall mit niedriger Latenz zu unterstützen.

- Amazon Elastic Compute Cloud (Amazon EC2) – Amazon EC2 bietet sichere und skalierbare Rechenkapazität in der AWS Cloud. Amazon EC2 reduziert die Notwendigkeit, im Voraus in Hardware investieren zu müssen. Daher können Sie Anwendungen schneller entwickeln und bereitstellen. Mit Amazon EC2 können Sie so viele oder so wenige virtuelle Server starten, wie Sie benötigen, die Sicherheit und das Netzwerk konfigurieren und den Speicher verwalten.
- AWS Lambda – Lambda ist ein Computingservice, mit dem Sie Code ausführen können, ohne Server bereitstellen oder verwalten zu müssen. Sie konfigurieren Benachrichtigungseinstellungen für einen Bucket und erteilen die Amazon-S3-Berechtigung zum Aufrufen einer Funktion in der ressourcenbasierten Berechtigungsrichtlinie der Funktion.
- Amazon Elastic Kubernetes Service (Amazon EKS) – Amazon EKS ist ein verwalteter Service, der die Installation, den Betrieb und die Wartung Ihrer eigenen Kubernetes Steuerebene auf überflüssig machtAWS. Kubernetes ist ein Open-Source-System, das die Verwaltung, Skalierung und Bereitstellung von containerisierten Anwendungen automatisiert.
- Amazon Elastic Container Service (Amazon ECS) – Amazon ECS ist ein vollständig verwalteter Container-Orchestrierungsservice, mit dem Sie containerisierte Anwendungen einfach bereitstellen, verwalten und skalieren können.
- Amazon Athena – Athena ist ein interaktiver Abfrageservice, der die Analyse von Daten in Amazon S3 mit Standard-SQL erleichtert. Sie können Athena auch verwenden, um Datenanalysen interaktiv auszuführen, indem Sie verwenden, Apache Spark ohne Ressourcen planen, konfigurieren oder verwalten zu müssen. Wenn Sie Apache Spark Anwendungen auf Athena ausführen, übermitteln Sie Spark Code zur Verarbeitung und erhalten die Ergebnisse direkt.
- Amazon SageMaker Runtime Model Training – Amazon SageMaker Runtime ist ein vollständig verwalteter Machine Learning-Service. Mit SageMaker Laufzeit können Datenwissenschaftler und Entwickler schnell und einfach Machine-Learning-Modelle erstellen und trainieren und diese dann direkt in einer produktionsbereiten gehosteten Umgebung bereitstellen.
- AWS Glue – AWS Glue ist ein Serverless-Datenintegrationsservice, der es Analysebenutzern erleichtert, Daten aus mehreren Quellen zu erkennen, vorzubereiten, zu verschieben und zu integrieren. Sie können AWS Glue für Analysen, Machine Learning und Anwendungsentwicklung verwenden. umfasst AWS Glue auch zusätzliche Produktivitäts- und Datenbetriebstools für die Erstellung, Ausführung von Aufträgen und die Implementierung von Geschäftsworkflows.
- Amazon EMR – Amazon EMR ist eine verwaltete Cluster-Plattform, die die Ausführung von Big-Data-Frameworks wie Apache Hadoop und vereinfachtApache Spark, AWS um riesige Datenmengen zu verarbeiten und zu analysieren.

Nächste Schritte

Weitere Informationen zum Arbeiten mit der Speicherklasse S3 Express One Zone und Verzeichnis-Buckets finden Sie in den folgenden Themen:

- Wodurch zeichnet sich S3 Express One Zone aus?
- Erste Schritte mit S3 Express One Zone
- Networking für S3 Express One Zone
- Verzeichnis-Buckets
- Arbeiten mit Objekten in einem Verzeichnis-Bucket
- Sicherheit für S3 Express One Zone
- Optimieren der Leistung von Amazon S3 Express One Zone
- Entwicklung mit S3 Express One Zone

Wodurch zeichnet sich S3 Express One Zone aus?

Amazon S3 Express One Zone ist eine leistungsstarke Amazon-S3-Speicherklasse mit einer einzelnen Zone, die speziell für den konsistenten Datenzugriff im einstelligen Millisekundenbereich für Ihre latenzempfindlichsten Anwendungen entwickelt wurde. S3 Express One Zone ist die erste S3-Speicherklasse, bei der Sie eine einzelne Availability Zone mit der Option auswählen können, Ihren Objektspeicher gemeinsam mit Ihren Computingressourcen zu platzieren, was die höchstmögliche Zugriffsgeschwindigkeit bietet. Um die Zugriffsgeschwindigkeit weiter zu erhöhen und Hunderttausende von Anfragen pro Sekunde zu unterstützen, werden S3-Express-One-Zone-Daten außerdem in einem neuen Bucket-Typ gespeichert: einem Amazon-S3-Verzeichnis-Bucket.

Weitere Informationen finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Sie können Verzeichnis-Buckets erstellen und mithilfe der Amazon-S3-API auf Ihre Daten in S3 Express One Zone zugreifen. Die Amazon-S3-API ist mit S3 Express One Zone und den Verzeichnis-Buckets kompatibel, mit Ausnahme einiger wichtiger Unterschiede. Weitere Informationen zu den Unterschieden von S3 Express One Zone finden Sie in den folgenden Themen.

Themen

- Unterschiede von S3 Express One Zone
- Von S3 Express One Zone unterstützte API-Operationen
- Amazon-S3-Features, die von S3 Express One Zone nicht unterstützt werden

Unterschiede von S3 Express One Zone

- Unterstützter Bucket-Typ – Objekte in Verzeichnis-Buckets werden in der Speicherklasse S3 Express One Zone gespeichert. Weitere Informationen finden Sie unter Verzeichnis-Buckets.
- Haltbarkeit – Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. S3 Express One Zone ist für eine Verfügbarkeit von 99,95 % innerhalb einer einzigen Availability Zone konzipiert und wird durch das Amazon S3 Service Level Agreement unterstützt. Weitere Informationen finden Sie unter Einzelne Availability Zone.
- **`ListObjectsV2`**-Verhalten – Bei Verzeichnis-Buckets gibt `ListObjectsV2` Objekte nicht in lexikographischer (alphabetischer) Reihenfolge zurück. Außerdem müssen Präfixe mit einem Trennzeichen enden und es kann nur „/“ als Trennzeichen angegeben werden.
- Löschverhalten – Wenn Sie ein Objekt in einem Verzeichnis-Bucket löschen, löscht Amazon S3 rekursiv alle leeren Verzeichnisse im Objektpfad. Wenn Sie beispielsweise den Objektschlüssel löschen`dir1/dir2/file1.txt`, löscht Amazon S3 `file1.txt`. Wenn die Verzeichnisse `dir1/` und `dir2/` leer sind und keine anderen Objekte enthalten, löscht Amazon S3 auch diese Verzeichnisse.
- ETags und Prüfsummen – Entity-Tags (ETags) für S3 Express One Zone sind zufällige alphanumerische Zeichenfolgen und keine MD5-Prüfsummen. Weitere Informationen zur Verwendung zusätzlicher Prüfsummen mit S3 Express One Zone finden Sie unter Bewährte Methoden für zusätzliche S3-Prüfsummen.
- Objektschlüssel in **`DeleteObjects`**-Anforderungen
  - Objektschlüssel in `DeleteObjects`-Anforderungen müssen mindestens ein Zeichen enthalten, das kein Leerzeichen ist. Zeichenfolgen, die ausschließlich Leerzeichen enthalten, werden in `DeleteObjects`-Anforderungen nicht unterstützt.
  - Objektschlüssel in `DeleteObjects`-Anforderungen dürfen keine Unicode-Steuerzeichen enthalten, mit Ausnahme von newline (`\n`), tab (`\t`) und carriage return (`\r`).
- Regionale und zonale Endpunkte – Wenn Sie S3 Express One Zone verwenden, müssen Sie die Region in allen Client-Anfragen angeben. Für regionale Endpunkte geben Sie die Region an, zum Beispiel `s3express-control.us-west-2.amazonaws.com`. Für zonale Endpunkte geben Sie sowohl die Region als auch die Availability Zone an, zum Beispiel `s3express-usw2-az1.us-west-2.amazonaws.com`. Weitere Informationen finden Sie unter Regionale und zonale Endpunkte.
- Mehrteilige Uploads – Wie bei anderen in Amazon S3 gespeicherten Objekten können Sie große Objekte, die in der Speicherklasse S3 Express One Zone gespeichert sind, mithilfe des

mehrteiligen Upload-Prozesses hochladen und kopieren. Im Folgenden sind jedoch einige Unterschiede aufgeführt, wenn Sie den mehrteiligen Upload-Prozess für Objekte verwenden, die in S3 Express One Zone gespeichert sind. Weitere Informationen finden Sie unter the section called "Verwenden von mehrteiligen Uploads mit Verzeichnis-Buckets".

- Das Objekterstellungsdatum ist das Abschlussdatum des mehrteiligen Uploads.
- Bei mehrteiligen Uploads müssen aufeinanderfolgende Teilenummern verwendet werden. Wenn Sie versuchen, eine mehrteilige Upload-Anforderung mit nicht aufeinanderfolgenden Teilenummern abzuschließen, generiert Amazon S3 einen HTTP-`400 (Bad Request)`-Fehler.
- Der Initiator eines mehrteiligen Uploads kann die Anforderung für den mehrteiligen Upload nur abbrechen, wenn ihm durch die entsprechende `s3express:CreateSession`-Berechtigung der Zugriff auf `AbortMultipartUpload` ausdrücklich gewährt wurde. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

Von S3 Express One Zone unterstützte API-Operationen

Die Speicherklasse Amazon S3 Express One Zone unterstützt sowohl regionale (Bucket-Ebene oder Steuerebene) als auch zonale (Objektebene oder Datenebene) Endpunkt-API-Operationen. Weitere Informationen finden Sie unter Networking für S3 Express One Zone und Endpunkte und Gateway-VPC-Endpunkte.

Regionale Endpunkt-API-Operationen

Die folgenden regionalen Endpunkt-API-Operationen werden für S3 Express One Zone unterstützt:

- CreateBucket
- DeleteBucket
- DeleteBucketPolicy
- GetBucketPolicy
- ListDirectoryBuckets
- PutBucketPolicy

Zonale Endpunkt-API-Operationen

Die folgenden zonalen Endpunkt-API-Operationen werden für S3 Express One Zone unterstützt:

- CreateSession

- CopyObject
- DeleteObject
- DeleteObjects
- GetObject
- GetObjectAttributes
- HeadBucket
- HeadObject
- ListObjectsV2
- PutObject
- AbortMultipartUpload
- CompleteMultiPartUpload
- CreateMultipartUpload
- ListMultipartUploads
- ListParts
- UploadPart
- UploadPartCopy

Amazon-S3-Features, die von S3 Express One Zone nicht unterstützt werden

Die folgenden Amazon-S3-Features werden von S3 Express One Zone nicht unterstützt:

- AWS CloudTrail-Ereignisse auf Datenebene
- Von AWS-verwaltete Richtlinien
- AWS PrivateLink für S3
- MD5-Prüfsummen
- Multi-Faktor Authentifizierung (MFA) aktivieren
- S3-Objektsperre
- Zahlung durch den Anforderer
- S3 Access Grants
- S3-Zugriffspunkte

- Bucket-Tags
- Amazon- CloudWatch Anforderungsmetriken
- S3-Ereignis-Benachrichtigungen
- S3-Lebenszyklus
- Multiregionale S3-Zugriffspunkte
- S3-Objekt-Lambda-Zugangspunkte
- S3-Versioning
- S3-Bestand
- S3-Replikation
- Objekt-Tags
- S3 Select
- Serverzugriffsprotokolle
- Statisches Website-Hosting
- S3 Storage Lens
- S3-Storage-Lens-Gruppen
- S3 Transfer Acceleration
- Serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS)
- Serverseitige Verschlüsselung mit Schlüsseln, die von AWS Key Management Service (AWS KMS) (SSE-KMS) verwaltet werden
- Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Erste Schritte mit S3 Express One Zone

Im folgenden Abschnitt werden die ersten Schritte mit der Verwendung der Speicherklasse Amazon S3 Express One Zone und der Verzeichnis-Buckets beschrieben. Weitere Informationen finden Sie unter Was ist S3 Express One Zone?.

Themen

- Einrichten von AWS Identity and Access Management (IAM) mit S3 Express One Zone
- Konfigurieren von Gateway-VPC-Endpunkten

- Arbeiten mit S3 Express One Zone über die S3-Konsole AWS CLI, und AWS SDKs

Einrichten von AWS Identity and Access Management (IAM) mit S3 Express One Zone

AWS Identity and Access Management (IAM) ist ein AWS-Service , mit dem Administratoren den Zugriff auf - AWS Ressourcen sicher steuern können. IAM-Administratoren steuern, wer authentifiziert (angemeldet) und autorisiert (im Besitz von Berechtigungen) ist, Amazon-S3-Ressourcen in S3 Express One Zone zu nutzen. Sie können IAM ohne zusätzliche Kosten nutzen.

Standardmäßig haben Benutzer keine Berechtigungen für Verzeichnis-Buckets und S3-Express-One-Zone-Vorgänge. Um Zugriffsberechtigungen für Verzeichnis-Buckets und S3-Express-One-Zone-Vorgänge zu gewähren, können Sie IAM verwenden, um Benutzer oder Rollen zu erstellen und diesen Identitäten Berechtigungen zuzuweisen.

Die ersten Schritte mit IAM finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone und Auf IAM-Identitäten basierende IAM-Richtlinien für S3 Express One Zone.

Konfigurieren von Gateway-VPC-Endpunkten

Für den Zugriff auf S3 Express One Zone verwenden Sie regionale und zonale Endpunkte, die sich von den standardmäßigen Amazon-S3-Endpunkten unterscheiden. Je nachdem, welchen Amazon-S3-API-Vorgang Sie verwenden, ist entweder ein regionaler oder ein zonaler Endpunkt erforderlich. Eine vollständige Liste der unterstützten API-Operationen nach Endpunkttyp finden Sie unter Von S3 Express One Zone unterstützte API-Operationen. Sie müssen über einen Gateway Virtual-Private-Cloud (VPC)-Endpunkt auf zonale und regionale Endpunkte zugreifen. Informationen zur Konfiguration von Gateway-Endpunkten finden Sie unter Networking für S3 Express One Zone.

Arbeiten mit S3 Express One Zone über die S3-Konsole AWS CLI, und AWS SDKs

Sie können mit der Speicherklasse S3 Express One Zone und Verzeichnis-Buckets arbeiten, indem Sie die - AWS SDKs , die Amazon S3-Konsole, AWS Command Line Interface (AWS CLI) und die Amazon S3-REST-API verwenden.

S3-Konsole

Um mit der Verwendung der S3-Konsole zu beginnen, führen Sie diese Schritte aus:

- Erstellen eines Verzeichnis-Buckets
- Leeren eines Verzeichnis-Buckets
- Löschen eines Verzeichnis-Buckets

AWS SDKs

S3 Express One Zone unterstützt die folgenden AWS SDKs:

- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java 2.x
- AWS SDK for JavaScript v3
- AWS SDK for .NET
- AWS SDK for PHP
- AWS SDK for Python (Boto3)
- AWS SDK for Ruby
- AWS SDK for Kotlin
- AWS SDK for Rust

Wenn Sie mit S3 Express One Zone arbeiten, empfehlen wir, die neueste Version der AWS -SDKs zu verwenden. Die unterstützten AWS SDKs für S3 Express One Zone übernehmen die Einrichtung, Aktualisierung und Beendigung von Sitzungen in Ihrem Namen. Das bedeutet, dass Sie sofort mit der Verwendung von API-Operationen beginnen können, nachdem Sie die AWS SDKs heruntergeladen und installiert und die erforderlichen IAM-Berechtigungen konfiguriert haben. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

Informationen zu den AWS SDKs , einschließlich deren Download und Installation, finden Sie unter Tools zum Erstellen auf AWS.

SDK AWS -Beispiele finden Sie im Folgenden:

- Erstellen eines Verzeichnis-Buckets
- Leeren eines Verzeichnis-Buckets
- Löschen eines Verzeichnis-Buckets

AWS Command Line Interface (AWS CLI)

Sie können die AWS Command Line Interface (AWS CLI) verwenden, um Verzeichnis-Buckets zu erstellen und unterstützte regionale und zonale Endpunkt-API-Operationen für S3 Express One Zone zu verwenden.

Informationen zu den ersten Schritten mit der AWS CLI finden Sie unter Erste Schritte mit der AWS CLI in der AWS CLI -Befehlsreferenz.

> **Note**
>
> Um Verzeichnis-Buckets mit den High-Level-`aws s3`Befehlen zu verwenden, aktualisieren Sie Ihre AWS CLI auf die neueste Version. Weitere Informationen zum Installieren und Konfigurieren der AWS CLI finden Sie unter Installieren oder Aktualisieren der neuesten Version der AWS CLI in der AWS CLI -Befehlsreferenz.

AWS CLI Beispiele finden Sie im Folgenden:

- Erstellen eines Verzeichnis-Buckets
- Leeren eines Verzeichnis-Buckets
- Löschen eines Verzeichnis-Buckets

Networking für S3 Express One Zone

Für den Zugriff auf Amazon-S3-Express-One-Zone-Objekte und Verzeichnis-Buckets verwenden Sie zonale Endpunkte, die sich von den standardmäßigen Amazon-S3-Endpunkten unterscheiden. Je nachdem, welchen S3-API-Vorgang Sie verwenden, ist entweder ein zonaler oder ein regionaler Endpunkt erforderlich. Eine vollständige Liste der API-Operationen nach Endpunkttyp finden Sie unter Von S3 Express One Zone unterstützte API-Operationen.

Sie können über Gateway Virtual-Private-Cloud (VPC)-Endpunkte auf zonale und regionale Endpunkte zugreifen. Informationen zur Konfiguration von Gateway-VPC-Endpunkten finden Sie unter the section called “Konfiguration von VPC-Gateway-Endpunkten”.

In den folgenden Themen werden die Netzwerkanforderungen für den Zugriff auf S3 Express One Zone über einen Gateway-VPC-Endpunkt beschrieben.

Themen

Endpunkte

Sie können über Gateway-VPC-Endpunkte von Ihrer VPC aus auf Objekte der Speicherklasse Amazon S3 Express One und Verzeichnis-Buckets zugreifen. S3 Express One Zone verwendet regionale und zonale API-Endpunkte. Je nachdem, welche Amazon-S3-API-Operation Sie verwenden, ist entweder ein regionaler oder ein zonaler Endpunkt erforderlich. Für die Nutzung von Gateway-Endpunkten fallen keine zusätzlichen Gebühren an.

API-Vorgänge auf Bucket-Ebene (Steuerebene) sind über einen regionalen Endpunkt verfügbar und werden als API-Vorgänge für regionale Endpunkte bezeichnet. Beispiele für API-Operationen für regionalen Endpunkte sind `CreateBucket` und `DeleteBucket`. Wenn Sie einen Verzeichnis-Bucket erstellen, wählen Sie eine einzelne Availability Zone aus, in der Ihr Verzeichnis-Bucket erstellt wird. Nachdem Sie einen Verzeichnis-Bucket erstellt haben, können Sie mithilfe von API-Operationen für zonale Endpunkte die Objekte in Ihrem Verzeichnis-Bucket hochladen und verwalten.

API-Operationen auf Objektebene (oder Datenebene) sind über zonale Endpunkte verfügbar und werden als API-Operationen für zonale Endpunkte bezeichnet. Beispiele für API-Operationen für zonale Endpunkte sind `CreateSession` und `PutObject`.

Die folgende Tabelle zeigt die regionalen und zonalen API-Endpunkte, die für jede Region und Availability Zone verfügbar sind.

Konfiguration von VPC-Gateway-Endpunkten

Gehen Sie wie folgt vor, um einen Gateway-Endpunkt zu erstellen, der eine Verbindung zu Objekten der Speicherklasse Amazon S3 Express One Zone und Verzeichnis-Buckets herstellt.

So konfigurieren Sie einen Gateway-VPC-Endpunkt

1. Öffnen Sie die Amazon-VPC-Konsole unter https://console.aws.amazon.com/vpc/.
2. Wählen Sie im Navigationsbereich Endpunkte aus.
3. Wählen Sie Endpunkt erstellen.
4. Erstellen Sie einen Namen für Ihren Endpunkt.
5. Wählen Sie für Servicekategorie die Option AWS-Services aus.

6. Fügen Sie für Services den Filter Typ=Gateway hinzu und wählen Sie dann das Optionsfeld neben com.amazonaws***region***.s3express.
7. Wählen Sie für VPC die VPC, in der der Endpunkt erstellt werden soll.
8. Wählen Sie für Route tables (Routing-Tabellen) die Routing-Tabellen, die von dem Endpunkt verwendet werden sollen. Amazon VPC fügt automatisch eine Route hinzu, die den für den Service bestimmten Datenverkehr auf die Netzwerkschnittstelle des Endpunkts verweist.
9. Wählen Sie für Richtlinie Vollzugriff, um alle Operationen aller Prinzipale auf allen Ressourcen über den VPC-Endpunkt zuzulassen. Wählen Sie andernfalls Benutzerdefiniert, um eine VPC-Endpunktrichtlinie anzufügen, die die Berechtigungen steuert, die Prinzipale zum Ausführen von Aktionen für Ressourcen über den VPC-Endpunkt haben.
10. (Optional) Sie fügen ein Tag hinzu, indem Sie Neues Tag hinzufügen auswählen und den Schlüssel und den Wert für das Tag eingeben.
11. Wählen Sie Endpunkt erstellen aus.

Nachdem Sie einen Gateway-Endpunkt erstellt haben, können Sie regionale API-Endpunkte und zonale API-Endpunkte verwenden, um auf Objekte in der Speicherklasse Amazon S3 Express One Zone und Verzeichnis-Buckets zuzugreifen.

Verzeichnis-Buckets

Es gibt zwei Arten von Amazon-S3-Buckets: Allzweck-Buckets und Verzeichnis-Buckets. Wählen Sie den Bucket-Typ, der am besten zu Ihren Anwendungs- und Leistungsanforderungen passt:

- Allzweck-Buckets sind der ursprüngliche S3-Bucket-Typ und werden für die meisten Anwendungsfälle und Zugriffsmuster empfohlen. Allzweck-Buckets ermöglichen auch Objekte, die in allen Speicherklassen gespeichert werden, mit Ausnahme von S3 Express One Zone.
- Verzeichnis-Buckets verwenden die Speicherklasse S3 Express One Zone. Dies wird empfohlen, wenn Ihre Anwendung leistungsempfindlich ist und von `PUT`- und `GET`-Latenzen im einstelligen Millisekundenbereich profitiert.

Verzeichnis-Buckets werden für Workloads oder leistungskritische Anwendungen verwendet, die eine konsistente Latenzzeit im einstelligen Millisekundenbereich erfordern. Verzeichnis-Buckets organisieren Daten hierarchisch in Verzeichnissen, im Gegensatz zu der flachen Speicherstruktur von Allzweck-Buckets. Es gibt keine Präfixbeschränkungen für Verzeichnis-Buckets und einzelne Verzeichnisse können horizontal skaliert werden.

Verzeichnis-Buckets verwenden die Speicherklasse S3 Express One Zone, die Daten auf mehreren Geräten innerhalb einer einzigen Availability Zone speichert, Daten jedoch nicht redundant über Availability Zones hinweg speichert. Wenn Sie einen Verzeichnis-Bucket erstellen, empfehlen wir Ihnen, eine AWS-Region und eine Availability Zone anzugeben, die sich lokal auf Ihren Amazon EC2-, Amazon Elastic Kubernetes Service- oder Amazon Elastic Container Service (Amazon ECS)-Compute-Instances befinden, um die Leistung zu optimieren.

Verzeichnis-Buckets speichern Objekte in der Speicherklasse S3 Express One Zone, die eine schnellere Verarbeitung von Daten innerhalb einer einzigen Availability Zone ermöglicht. Weitere Informationen finden Sie unter Verzeichnis-Buckets.

Sie können bis zu 10 Verzeichnis-Buckets in jedem Ihrer erstellenAWS-Konten, ohne Begrenzung für die Anzahl der Objekte, die Sie in einem Bucket speichern können. Ihr Bucket-Kontingent wird au jede Region in Ihrem AWS-Konto angewendet. Wenn Ihre Anwendung eine Erhöhung dieses Limits erfordert, wenden Sie sich an AWS Support. Weitere Informationen finden Sie in der Service Quotas-Konsole.

> **Important**
>
> Verzeichnis-Buckets, die für einen Zeitraum von mindestens 90 Tagen keine Anforderungsaktivität aufweisen, wechseln in einen inaktiven Zustand. Im inaktiven Zustand ist ein Verzeichnis-Bucket vorübergehend für Lese- und Schreibvorgänge nicht zugänglich. Inaktive Buckets behalten alle Speicher-, Objekt- und Bucket-Metadaten bei. Bestehende Speichergebühren gelten für inaktive Buckets. Wenn Sie eine Zugriffsanfrage an einen inaktiven Bucket stellen, wechselt der Bucket in einen aktiven Zustand, in der Regel innerhalb weniger Minuten. Während dieses Übergangszeitraums geben Lese- und Schreibvorgänge einen HTTP-`503 (Service Unavailable)`Fehlercode zurück.

Die folgenden Themen enthalten Informationen zu Verzeichnis-Buckets. Weitere Informationen über Allzweck-Buckets finden Sie unter Bucket-Übersicht.

Themen

- Availability Zones
- Namen von Verzeichnis-Buckets
- Verzeichnisse
- Schlüsselnamen

- Zugriffsverwaltung
- Arbeiten mit Verzeichnis-Buckets
- Regeln für die Benennung von Verzeichnis-Buckets
- Erstellen eines Verzeichnis-Buckets
- Anzeigen von Verzeichnis-Bucket-Eigenschaften
- Verwalten von Bucket-Richtlinien für Verzeichnis-Buckets
- Leeren eines Verzeichnis-Buckets
- Löschen eines Verzeichnis-Buckets
- Auflisten von Verzeichnis-Buckets
- Verwenden von HeadBucket mit Verzeichnis-Buckets

Availability Zones

Wenn Sie einen Verzeichnis-Bucket erstellen, wählen Sie die Availability Zone und die AWS-Region.

Verzeichnis-Buckets verwenden die Speicherklasse S3 Express One Zone, die für die Verwendung durch leistungssensitive Anwendungen konzipiert wurde. S3 Express One Zone ist die erste S3-Speicherklasse, bei der Sie eine einzelne Availability Zone mit der Option auswählen können, Ihren Objektspeicher gemeinsam mit Ihren Computingressourcen zu platzieren, was die höchstmögliche Zugriffsgeschwindigkeit bietet.

Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. S3 Express One Zone ist für eine Verfügbarkeit von 99,95 Prozent innerhalb einer einzigen Availability Zone konzipiert und wird durch das Amazon S3 Service Level Agreement unterstützt. Weitere Informationen finden Sie unter Einzelne Availability Zone.

Namen von Verzeichnis-Buckets

Der Name eines Verzeichnis-Buckets besteht aus einem Basisnamen, den Sie angeben, und einem Suffix, das die ID der Availability Zone enthält, in der sich Ihr Bucket befindet. Die Namen von Verzeichnis-Buckets müssen das folgende Format haben und den Benennungsregeln für Verzeichnis-Buckets entsprechen:

```
bucket-base-name--azid--x-s3
```

Der folgende Verzeichnis-Bucket-Name enthält beispielsweise die Availability Zone-ID `usw2-az1`:

```
bucket-base-name--usw2-az1--x-s3
```

Weitere Informationen finden Sie unter Regeln für die Benennung von Verzeichnis-Buckets.

Verzeichnisse

Verzeichnis-Buckets organisieren Daten hierarchisch in Verzeichnissen, im Gegensatz zu der flachen Sortierstruktur von Allzweck-Buckets. Jeder S3-Verzeichnis-Bucket kann Hunderttausende von Transaktionen pro Sekunde (TPS) unterstützen, unabhängig von der Anzahl der Verzeichnisse innerhalb des Buckets.

Bei einem hierarchischen Namespace ist das Trennzeichen im Objektschlüssel wichtig. Das einzige unterstützte Trennzeichen ist der Schrägstrich (`/`). Verzeichnisse werden durch Trennzeichengrenzen bestimmt. Beispielsweise führt der Objektschlüssel `dir1/dir2/file1.txt` dazu, dass die Verzeichnisse `dir1`/und `dir2/` automatisch erstellt werden und das Objekt `file1.txt` dem `/dir2`-Verzeichnis im Pfad `dir1/dir2/file1.txt` hinzugefügt wird.

Das Indizierungsmodell für Verzeichnis-Buckets gibt unsortierte Ergebnisse für den `ListObjectsV2`-API-Vorgang zurück. Wenn Sie Ihre Ergebnisse auf einen Unterabschnitt Ihres Buckets beschränken müssen, können Sie im `prefix`-Parameter einen Unterverzeichnispfad angeben, z. B. `prefix=dir1/`.

Schlüsselnamen

Bei Verzeichnis-Buckets werden Unterverzeichnisse, die mehreren Objektschlüsseln gemeinsam sind, mit dem ersten Objektschlüssel erstellt. Zusätzliche Objektschlüssel für dasselbe Unterverzeichnis verwenden das zuvor erstellte Unterverzeichnis. Dieses Modell bietet Ihnen Flexibilität bei der Auswahl von Objektschlüsseln, die für die Anwendung am besten geeignet sind, und unterstützt sowohl dünn besetzte als auch dichte Verzeichnisse.

Zugriffsverwaltung

Bei Verzeichnis-Buckets sind alle Einstellungen für das Blockieren des öffentlichen Zugriffs auf Bucket-Ebene standardmäßig aktiviert. Die S3-Objekt-Eigentümerschaft ist auf „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle Zugriffssteuerungslisten sind deaktiviert. Diese Einstellungen können nicht geändert werden.

Standardmäßig haben Benutzer keine Berechtigungen für Verzeichnis-Buckets und S3-Express-One-Zone-Vorgänge. Um Zugriffsberechtigungen für Verzeichnis-Buckets zu gewähren, können Sie IAM verwenden, um Benutzer, Gruppen oder Rollen zu erstellen und diesen Identitäten Berechtigungen zuzuweisen. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

Arbeiten mit Verzeichnis-Buckets

Weitere Informationen zum Arbeiten mit Verzeichnis-Buckets finden Sie in den folgenden Themen.

Themen

- Regeln für die Benennung von Verzeichnis-Buckets
- Erstellen eines Verzeichnis-Buckets
- Anzeigen von Verzeichnis-Bucket-Eigenschaften
- Verwalten von Bucket-Richtlinien für Verzeichnis-Buckets
- Leeren eines Verzeichnis-Buckets
- Löschen eines Verzeichnis-Buckets
- Auflisten von Verzeichnis-Buckets
- Verwenden von HeadBucket mit Verzeichnis-Buckets

Regeln für die Benennung von Verzeichnis-Buckets

Wenn Sie einen Verzeichnis-Bucket in Amazon S3 erstellen, gelten dafür die folgenden Benennungsregeln. Benennungsregeln für Allzweck-Buckets finden Sie unterRegeln für die Benennung von Buckets.

Ein Verzeichnis-Bucket-Name besteht aus einem Basisnamen, den Sie angeben, und einem Suffix, das die ID der AWS Availability Zone enthält, in der sich Ihr Bucket befindet, und `--x-s3`.

```
base-name--azid--x-s3
```

Der folgende Verzeichnis-Bucket-Name enthält beispielsweise die Availability Zone-ID `usw2-az1`:

```
bucket-base-name--usw2-az1--x-s3
```

> **Note**
>
> Wenn Sie einen Verzeichnis-Bucket mit der Konsole erstellen, wird dem von Ihnen angegebenen Basisnamen automatisch ein Suffix hinzugefügt. Dieses Suffix enthält die ID der Availability Zone, die Sie ausgewählt haben.
> Wenn Sie einen Verzeichnis-Bucket mit einer API erstellen, müssen Sie das vollständige Suffix, einschließlich der Availability Zone-ID, in Ihrer Anfrage angeben. Eine Liste der Availability Zone-IDs finden Sie unter Availability Zones und Regionen bei S3 Express One Zone.

Die Namen von Verzeichnis-Buckets müssen folgende Kriterien erfüllen:

- innerhalb der ausgewählten AWS-Region und Availability Zone eindeutig sein.
- Nicht mehr als 3-63 Zeichen lang sein, einschließlich des Suffixes.
- Nur aus Kleinbuchstaben, Zahlen und Bindestrichen bestehen.
- Muss mit einer Zahl oder einem Buchstaben beginnen und enden.
- Muss das folgende Suffix enthalten: `--azid--x-s3`.

Erstellen eines Verzeichnis-Buckets

Um mit der Verwendung der Speicherklasse Amazon S3 Express One Zone zu beginnen, erstellen Sie einen Verzeichnis-Buckets. Die Speicherklasse S3 Express One Zone kann nur mit Verzeichnis-Buckets verwendet werden. Die Speicherklasse S3 Express One Zone unterstützt Anwendungsfälle mit niedriger Latenz und ermöglicht eine schnellere Datenverarbeitung innerhalb einer einzigen Availability Zone. Wenn Ihre Anwendung leistungsempfindlich ist und von PUT- und GET-Latenzen im einstelligen Millisekundenbereich profitiert, empfehlen wir, einen Verzeichnis-Bucket zu erstellen, damit Sie die Speicherklasse S3 Express One Zone verwenden können.

Es gibt zwei Arten von Amazon-S3-Buckets: Allzweck-Buckets und Verzeichnis-Buckets. Sie sollten den Bucket-Typ wählen, der am besten zu Ihren Anwendungs- und Leistungsanforderungen passt. Bei Allzweck-Buckets handelt es sich um den ursprünglichen S3-Bucket-Typ. Allzweck-Buckets werden für die meisten Anwendungsfälle und Zugriffsmuster empfohlen und erlauben Objekte, die in allen Speicherklassen gespeichert sind, außer S3 Express One Zone. Weitere Informationen über Allzweck-Buckets finden Sie unter Bucket-Übersicht.

Verzeichnis-Buckets verwenden die Speicherklasse S3 Express One Zone, die für Workloads oder leistungskritische Anwendungen konzipiert ist, die eine konsistente Latenz im einstelligen Millisekundenbereich erfordern. S3 Express One Zone ist die erste S3-Speicherklasse, bei der Sie eine einzelne Availability Zone mit der Option auswählen können, Ihren Objektspeicher gemeinsam mit Ihren Computingressourcen zu platzieren, was die höchstmögliche Zugriffsgeschwindigkeit bietet. Wenn Sie einen Verzeichnis-Bucket erstellen, können Sie optional eine AWS-Region und eine Availability Zone angeben, die sich lokal auf Ihren Amazon EC2-, Amazon Elastic Kubernetes Service- oder Amazon Elastic Container Service (Amazon ECS)-Compute-Instances befinden, um die Leistung zu optimieren.

Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. S3 Express One Zone ist für eine Verfügbarkeit von 99,95 Prozent innerhalb einer einzigen Availability Zone konzipiert und wird durch das Amazon S3 Service Level Agreement unterstützt. Weitere Informationen finden Sie unter Einzelne Availability Zone.

Verzeichnis-Buckets organisieren Daten hierarchisch in Verzeichnissen, im Gegensatz zur flachen Speicherstruktur von Allzweck-Buckets. Es gibt keine Präfixbeschränkungen für Verzeichnis-Buckets und einzelne Verzeichnisse können horizontal skaliert werden.

Weitere Information zu Verzeichnis-Buckets finden Sie unter Verzeichnis-Buckets.

Namen von Verzeichnis-Buckets

Die Namen von Verzeichnis-Buckets müssen diesem Format folgen und den Regeln für die Benennung von Verzeichnis-Buckets entsprechen:

```
bucket-base-name--azid--x-s3
```

Der folgende Verzeichnis-Bucket-Name enthält beispielsweise die Availability Zone-ID `usw2-az1`:

```
bucket-base-name--usw2-az1--x-s3
```

Weitere Informationen zu den Benennungsregeln für Verzeichnis-Buckets finden Sie unter Regeln für die Benennung von Verzeichnis-Buckets.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Anschließend wird die Seite Bucket erstellen geöffnet.
4. Wählen Sie für Region die aus, AWS-Region in der sich der Verzeichnis-Bucket befinden soll.

   Um Latenz und Kosten zu minimieren und gesetzliche Anforderungen zu erfüllen, wählen Sie eine Region in Ihrer Nähe aus. In einer Region gespeicherte Objekte verbleiben so lange in der Region, bis sie explizit in eine andere Region verschoben werden. Eine Liste von Amazon S3 AWS-Regionen finden Sie unter -AWS-Service Endpunkte im Allgemeine Amazon Web Services-Referenz.
5. Wählen Sie für Bucket-Typ die Option Verzeichnis aus.

   > **Note**
   >
   > - Wenn Sie eine Region ausgewählt haben, die keine Verzeichnis-Buckets unterstützt, verschwindet die Option Bucket-Typ und der Bucket-Typ ist standardmäßig ein Allzweck-Bucket. Um einen Verzeichnis-Bucket zu erstellen, müssen Sie eine unterstützte Region auswählen. Eine Liste der Regionen, die Verzeichnis-Buckets unterstützen, und der Speicherklasse Amazon S3 Express One Zone finden Sie unter the section called “Availability Zones und Regionen bei S3 Express One Zone”.
   > - Nach dem Erstellen des Buckets kann der Bucket-Typ nicht mehr geändert werden.

6. Wählen Sie für Availability Zone eine Availability Zone aus, die für Ihre Computing-Services lokal ist. Eine Liste der Availability Zones, die Verzeichnis-Buckets unterstützen, und der Speicherklasse S3 Express One Zone finden Sie unter the section called “Availability Zones und Regionen bei S3 Express One Zone”.

   > **Note**
   >
   > Die Availability Zone kann nicht geändert werden, nachdem der Bucket erstellt wurde.

7. Markieren Sie unter Availability Zone das Kontrollkästchen, um zu bestätigen, dass Ihre Daten im Falle eines Ausfalls der Availability Zone möglicherweise nicht verfügbar sind oder verloren gehen.

> **Important**
>
> Obwohl Verzeichnis-Buckets auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert werden, speichern Verzeichnis-Buckets Daten nicht redundant über Availability Zones hinweg.

8. Geben Sie unter Bucket-Name einen Namen für Ihren Verzeichnis-Bucket ein.

   Die Namen von Verzeichnis-Buckets müssen folgende Kriterien erfüllen:

   - innerhalb der ausgewählten AWS-Region und Availability Zone eindeutig sein.
   - Nicht mehr als 3-63 Zeichen lang sein, einschließlich des Suffixes.
   - Nur aus Kleinbuchstaben, Zahlen und Bindestrichen bestehen.
   - Muss mit einer Zahl oder einem Buchstaben beginnen und enden.
   - Muss das folgende Suffix enthalten: `--azid--x-s3`.

   Dem Basisnamen, den Sie beim Erstellen eines Verzeichnis-Buckets mit der Konsole angeben, wird automatisch ein Suffix hinzugefügt. Dieses Suffix enthält die ID der Availability Zone, die Sie ausgewählt haben.

   Der Name eines einmal erstellter Buckets kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

   > **Important**
   >
   > Geben Sie keine sensiblen Informationen wie Kontonummern in den Bucket-Namen ein. Der Bucket-Name wird in der URL angezeigt, die auf die Objekte im Bucket verweist.

9. Unter Object Ownership wird die Einstellung „Bucket-Eigentümer erzwungen“ automatisch aktiviert und alle Zugriffssteuerungslisten (ACLs) sind deaktiviert. Für Verzeichnis-Buckets können ACLs nicht aktiviert werden.

   Deaktivierte ACLs

   - Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben

keine Auswirkungen mehr auf Zugriffsberechtigungen für Daten im S3-Bucket. Der Bucket verwendet ausschließlich Richtlinien, um die Zugriffssteuerung zu definieren.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

10. Unter Block Public Access-Einstellungen für diesen Bucket werden alle Block Public Access-Einstellungen für Ihren Verzeichnis-Bucket automatisch aktiviert. Diese Einstellungen können für Verzeichnis-Buckets nicht geändert werden. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.
11. Unter Serverseitige Verschlüsselungseinstellungen wendet Amazon S3 die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) als Basisverschlüsselungsstufe für alle S3-Buckets an. Alle Objekt-Uploads in Verzeichnis-Buckets werden mit SSE-S3 verschlüsselt. Bei Verzeichnis-Buckets kann der Verschlüsselungstyp nicht geändert werden. Weitere Informationen zu SSE-S3 finden Sie unter the section called “Von Amazon S3 verwaltete Verschlüsselungsschlüssel (SSE-S3)”.
12. Wählen Sie Bucket erstellen aus.

    Nachdem Sie den Bucket erstellt haben, können Sie ihm Dateien und Ordner hinzufügen. Weitere Informationen finden Sie unter the section called “Arbeiten mit Objekten in einem Verzeichnis-Bucket”.

Verwenden der AWS SDKs

SDK for Go

Dieses Beispiel zeigt, wie Sie einen Verzeichnis-Bucket mithilfe der erstellen AWS SDK for Go.

Example

```
var bucket = "..."

func runCreateBucket(c *s3.Client) {
    resp, err := c.CreateBucket(context.Background(), &s3.CreateBucketInput{
        Bucket: &bucket,
        CreateBucketConfiguration: &types.CreateBucketConfiguration{
            Location: &types.LocationInfo{
                Name: aws.String("usw2-az1"),
```

```
                Type: types.LocationTypeAvailabilityZone,
            },
            Bucket: &types.BucketInfo{
                DataRedundancy: types.DataRedundancySingleAvailabilityZone,
                Type:           types.BucketTypeDirectory,
            },
        },
    })
    var terr *types.BucketAlreadyOwnedByYou
    if errors.As(err, &terr) {
        fmt.Printf("BucketAlreadyOwnedByYou: %s\n", aws.ToString(terr.Message))
        fmt.Printf("noop...\n")
        return
    }
    if err != nil {
        log.Fatal(err)
    }

    fmt.Printf("bucket created at %s\n", aws.ToString(resp.Location))
}
```

SDK for Java 2.x

Dieses Beispiel zeigt, wie Sie einen -Verzeichnis-Bucket mithilfe der erstellen AWS SDK for Java 2.x.

Example

```
public static void createBucket(S3Client s3Client, String bucketName) {

    //Bucket name format is {base-bucket-name}--{az-id}--x-s3
    //example: doc-example-bucket--usw2-az1--x-s3 is a valid name for a directory
 bucket created in
    //Region us-west-2, Availability Zone 2

    CreateBucketConfiguration bucketConfiguration =
 CreateBucketConfiguration.builder()
            .location(LocationInfo.builder()
                    .type(LocationType.AVAILABILITY_ZONE)
                    .name("usw2-az1").build()) //this must match the Region and
 Availability Zone in your bucket name
            .bucket(BucketInfo.builder()
                   .type(BucketType.DIRECTORY)
```

```
                    .dataRedundancy(DataRedundancy.SINGLE_AVAILABILITY_ZONE)
                    .build()).build();
    try {

             CreateBucketRequest bucketRequest =
 CreateBucketRequest.builder().bucket(bucketName).createBucketConfiguration(bucketConfigurat
             CreateBucketResponse response = s3Client.createBucket(bucketRequest);
             System.out.println(response);
    }

    catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
         }
    }
```

AWS SDK for JavaScript

Dieses Beispiel zeigt, wie Sie einen Verzeichnis-Bucket mithilfe der erstellen AWS SDK for JavaScript.

Example

```
// file.mjs, run with Node.js v16 or higher
// To use with the preview build, place this in a folder
// inside the preview build directory, such as /aws-sdk-js-v3/workspace/

import { S3 } from "@aws-sdk/client-s3";

const region = "us-east-1";
const zone = "use1-az4";
const suffix = `${zone}--x-s3`;

const s3 = new S3({ region });

const bucketName = `...--${suffix}`;

const createResponse = await s3.createBucket(
    { Bucket: bucketName,
      CreateBucketConfiguration: {Location: {Type: "AvailabilityZone", Name: zone},
      Bucket: { Type: "Directory", DataRedundancy: "SingleAvailabilityZone" }}
```

```
    }
   );
```

AWS SDK for .NET

Dieses Beispiel zeigt, wie Sie einen Verzeichnis-Bucket mithilfe der erstellen AWS SDK for .NET.

Example

```
using (var amazonS3Client = new AmazonS3Client())
{
    var putBucketResponse = await amazonS3Client.PutBucketAsync(new PutBucketRequest
    {

       BucketName = "DOC-EXAMPLE-BUCKET--usw2-az1--x-s3",
       PutBucketConfiguration = new PutBucketConfiguration
       {
         BucketInfo = new BucketInfo { DataRedundancy =
 DataRedundancy.SingleAvailabilityZone, Type = BucketType.Directory },
         Location = new LocationInfo { Name = "usw2-az1", Type =
 LocationType.AvailabilityZone }
       }
     }).ConfigureAwait(false);
}
```

SDK for PHP

Dieses Beispiel zeigt, wie Sie einen Verzeichnis-Bucket mithilfe der erstellen AWS SDK for PHP.

Example

```
require 'vendor/autoload.php';

$s3Client = new S3Client([

    'region'      => 'us-east-1',
]);


$result = $s3Client->createBucket([
    'Bucket' => 'doc-example-bucket--use1-az4--x-s3',
    'CreateBucketConfiguration' => [
```

```
        'Location' => ['Name'=> 'use1-az4', 'Type'=> 'AvailabilityZone'],
        'Bucket' => ["DataRedundancy" => "SingleAvailabilityZone" ,"Type" =>
 "Directory"]   ],
]);
```

SDK for Python

Dieses Beispiel zeigt, wie Sie einen Verzeichnis-Bucket mithilfe der erstellen AWS SDK for Python (Boto3).

Example

```
import logging
import boto3
from botocore.exceptions import ClientError

def create_bucket(s3_client, bucket_name, availability_zone):
    '''
    Create a directory bucket in a specified Availability Zone

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket to create; for example, 'doc-example-bucket--usw2-
az1--x-s3'
    :param availability_zone: String; Availability Zone ID to create the bucket in,
 for example, 'usw2-az1'
    :return: True if bucket is created, else False
    '''

    try:
        bucket_config = {
                'Location': {
                    'Type': 'AvailabilityZone',
                    'Name': availability_zone
                },
                'Bucket': {
                    'Type': 'Directory',
                    'DataRedundancy': 'SingleAvailabilityZone'
                }
            }
        s3_client.create_bucket(
            Bucket = bucket_name,
            CreateBucketConfiguration = bucket_config
        )
```

```
    except ClientError as e:
        logging.error(e)
        return False
    return True


if __name__ == '__main__':
    bucket_name = 'BUCKET_NAME'
    region = 'us-west-2'
    availability_zone = 'usw2-az1'
    s3_client = boto3.client('s3', region_name = region)
    create_bucket(s3_client, bucket_name, availability_zone)
```

SDK for Ruby

Dieses Beispiel zeigt, wie Sie einen -Verzeichnis-Bucket mithilfe der erstellen AWS SDK for Ruby.

Example

```
s3 = Aws::S3::Client.new(region:'us-west-2')
s3.create_bucket(
  bucket: "bucket_base_name--az_id--x-s3",
  create_bucket_configuration: {
    location: { name: 'usw2-az1', type: 'AvailabilityZone' },
    bucket: { data_redundancy: 'SingleAvailabilityZone', type: 'Directory' }
  }
)
```

Anzeigen von Verzeichnis-Bucket-Eigenschaften

Sie können die Eigenschaften für einen Amazon S3-Verzeichnis-Bucket mithilfe der Amazon S3-Konsole anzeigen und konfigurieren. Weitere Informationen finden Sie unter Verzeichnis-Buckets und Was ist S3 Express One Zone?.

Verwenden der S3-Konsole

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie die Registerkarte Verzeichnis-Buckets.

4. Wählen Sie in der Liste Verzeichnis-Buckets den Namen des Buckets aus, dessen Eigenschaften Sie anzeigen möchten.
5. Wählen Sie die Registerkarte Eigenschaften aus.
6. Auf der Registerkarte Eigenschaften können Sie die folgenden Eigenschaften für den Bucket anzeigen:

   - Verzeichnis-Bucket-Übersicht – Sie können die AWS-Region, die Availability Zone, den Amazon-Ressourcennamen (ARN) und das Erstellungsdatum für den Bucket sehen.
   - Standard-Verschlüsselung – Amazon S3 wendet eine serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) als Basisverschlüsselungsstufe für alle S3-Buckets an. Für Verzeichnis-Buckets kann diese Einstellung nicht geändert werden. Amazon S3 verschlüsselt ein Objekt, bevor es auf einer Festplatte gespeichert wird, und entschlüsselt das Objekt beim Herunterladen. Weitere Informationen finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

     Weitere Informationen zu unterstützten Features für Verzeichnis-Buckets finden Sie unter Funktionen von S3 Express One Zone.

Verwalten von Bucket-Richtlinien für Verzeichnis-Buckets

Sie können Bucket-Richtlinien für Amazon S3-Verzeichnis-Buckets mithilfe der Amazon S3-Konsole und der AWS SDKs hinzufügen, löschen, aktualisieren und anzeigen. SDKs Weitere Informationen finden Sie unter den folgenden Themen. Weitere Informationen zu unterstützten AWS Identity and Access Management (IAM)-Aktionen und Bedingungsschlüsseln für S3 Express One Zone finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone. Beispiele für Bucket-Richtlinien für Verzeichnis-Buckets finden Sie unter Beispiel für Verzeichnis-Bucket-Richtlinien für S3 Express One Zone.

Themen

- Hinzufügen einer Bucket-Richtlinie
- Löschen einer Bucket-Richtlinie

Hinzufügen einer Bucket-Richtlinie

Um einem Verzeichnis-Bucket eine Bucket-Richtlinie hinzuzufügen, können Sie die Amazon S3-Konsole oder die -AWSSDKs verwenden. SDKs

Verwenden der S3-Konsole

Eine Bucket-Richtlinie erstellen oder bearbeiten

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie die Registerkarte Verzeichnis-Buckets.
4. Wählen Sie in der Liste Verzeichnis-Buckets den Namen des Buckets aus, in den Ihre Ordner oder Dateien hochgeladen werden sollen.
5. Wählen Sie die Registerkarte Berechtigungen.
6. Wählen Sie unter Bucket-Richtlinie Bearbeiten aus. Die Seite Bucket-Richtlinie bearbeiten wird angezeigt.
7. Um eine Richtlinie automatisch zu generieren, wählen Sie Richtliniengenerator aus.

   Wenn Sie Policy generator (Richtliniengenerator) auswählen, wird der AWS-Richtliniengenerator in einem neuen Fenster geöffnet.

   Wenn Sie den AWS Richtliniengenerator nicht verwenden möchten, können Sie JSON-Anweisungen im Abschnitt Richtlinie hinzufügen oder bearbeiten.

   a. Wählen Sie auf der Seite AWS Policy Generator (Richtliniengenerator) unter Select Type of Policy (Richtlinientyp auswählen) die Option S3 Bucket Policy (S3-Bucket-Richtlinie) aus.

   b. Fügen Sie eine Anweisung hinzu, indem Sie die Informationen in die bereitgestellten Felder eingeben, und wählen Sie dann Anweisung hinzufügen. Wiederholen Sie diesen Schritt für so viele Anweisungen, wie Sie hinzufügen möchten. Weitere Informationen zu diesen Feldern finden Sie in der Referenz zu den IAM-JSON-Richtlinienelementen im IAM-Benutzerhandbuch.

   > **Note**
   >
   > Der Einfachheit halber wird auf der Seite Bucket-Richtlinie bearbeiten der Bucket-ARN (Amazon-Ressourcenname) des aktuellen Buckets über dem Textfeld Richtlinie angezeigt. Sie können diesen ARN zur Verwendung in den Anweisungen auf der Seite AWS-Richtliniengenerator kopieren.

c. Wenn Sie mit dem Hinzufügen von Anweisungen fertig sind, wählen Sie Generieren von Richtlinien.

d. Kopieren Sie den generierten Richtlinientext, wählen Sie Schließen und kehren Sie zur Seite Bucket-Richtlinie bearbeiten in der Amazon-S3-Konsole zurück.

8. Bearbeiten Sie im Feld Policy (Richtlinie) die vorhandene Richtlinie oder fügen Sie die Bucket-Richtlinie aus dem AWS-Richtliniengenerator ein. Beheben Sie Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge bevor Sie Ihre Richtlinie speichern.

> **Note**
>
> Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt.

9. Wählen Sie Save changes (Änderungen speichern) aus, wodurch Sie zur Registerkarte Permissions (Berechtigungen) zurückkehren.

Verwenden der AWS-SDKs

SDK for Java 2.x

Example

`PutBucketPolicy` AWS SDK for Java 2.x

```
public static void setBucketPolicy(S3Client s3Client, String bucketName, String
 policyText) {

      //sample policy text
      /**
       * policy_statement = {
        *         'Version': '2012-10-17',
        *         'Statement': [
        *             {
        *                 'Sid': 'AdminPolicy',
        *                 'Effect': 'Allow',
        *                 'Principal': {
        *                     "AWS": "111122223333"
        *                 },
        *                 'Action': 's3express:*',
        *                 'Resource':
 'arn:aws:s3express:region:111122223333:bucket/bucket-base-name--azid--x-s3'
```

```
        *             }
        *         ]
        *     }
        */
        System.out.println("Setting policy:");
        System.out.println("----");
        System.out.println(policyText);
        System.out.println("----");
        System.out.format("On Amazon S3 bucket: \"%s\"\n", bucketName);

        try {
            PutBucketPolicyRequest policyReq = PutBucketPolicyRequest.builder()
                    .bucket(bucketName)
                    .policy(policyText)
                    .build();
            s3Client.putBucketPolicy(policyReq);
            System.out.println("Done!");
        }

        catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

Löschen einer Bucket-Richtlinie

Verwenden Sie die folgenden AWS SDK-Beispiele, um eine Bucket-Richtlinie für einen Verzeichnis-Bucket zu löschen.

Verwenden der AWS-SDKs

SDK for Java 2.x

Example

`DeleteBucketPolicy` AWS SDK for Java 2.x

```
public static void deleteBucketPolicy(S3Client s3Client, String bucketName) {
      try {
          DeleteBucketPolicyRequest deleteBucketPolicyRequest =
 DeleteBucketPolicyRequest
```

```
                .builder()
                .bucket(bucketName)
                .build()
        s3Client.deleteBucketPolicy(deleteBucketPolicyRequest);
        System.out.println("Successfully deleted bucket policy");
    }

    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
```

Leeren eines Verzeichnis-Buckets

Sie können einen Amazon S3-Verzeichnis-Bucket mithilfe der Amazon S3-Konsole leeren. Weitere Information zu Verzeichnis-Buckets finden Sie unter Verzeichnis-Buckets.

Bevor Sie einen Verzeichnis-Bucket leeren, beachten Sie Folgendes:

- Wenn Sie einen Verzeichnis-Bucket leeren, löschen Sie alle Objekte, behalten aber den Bucket selbst.
- Nachdem Sie einen Verzeichnis-Bucket geleert haben, kann die leere Aktion nicht mehr rückgängig gemacht werden.
- Objekte, die dem Verzeichnis-Bucket hinzugefügt werden, während die Aktion zum Leeren des Buckets ausgeführt wird, können gelöscht werden.

Wenn Sie den Bucket auch löschen möchten, beachten Sie Folgendes:

- Alle Objekte in einem Bucket müssen gelöscht werden, bevor der Bucket selbst gelöscht werden kann.
- Laufende mehrteilige Uploads in dem Verzeichnis-Bucket müssen abgebrochen werden, bevor der Bucket selbst gelöscht werden kann.

Informationen zum Löschen eines Verzeichnis-Buckets finden Sie unter Löschen eines Verzeichnis-Buckets. Informationen zum Abbrechen eines laufenden mehrteiligen Uploads finden Sie unter the section called “Abbrechen eines mehrteiligen Uploads”.

Informationen zum Leeren eines Allzweck-Buckets finden Sie unter Leeren eines Buckets.

Verwenden der S3-Konsole

So leeren Sie einen Verzeichnis-Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie die Registerkarte Verzeichnis-Buckets.
4. Wählen Sie das Optionsfeld neben dem Namen des Buckets aus, den Sie leeren möchten, und wählen Sie dann Leeren aus.
5. Bestätigen Sie auf der Seite Empty bucket (Bucket leeren), dass Sie den Bucket leeren möchten, indem Sie **`permanently delete`** in das Textfeld eingeben und dann Empty (Leeren) auswählen.
6. Überwachen Sie den Fortschritt des Bucket-Entleerungsvorgangs auf der Seite Leerer Bucket: Status.

Löschen eines Verzeichnis-Buckets

Sie können nur leere Amazon S3-Verzeichnis-Buckets löschen. Bevor Sie Ihren Verzeichnis-Bucket löschen, müssen Sie alle Objekte im Bucket löschen und alle laufenden mehrteiligen Uploads abbrechen.

Informationen zum Leeren eines Verzeichnis-Buckets finden Sie unter Leeren eines Verzeichnis-Buckets. Informationen zum Abbrechen eines laufenden mehrteiligen Uploads finden Sie unter the section called “Abbrechen eines mehrteiligen Uploads”.

Informationen zum Löschen eines Allzweck-Buckets finden Sie unter Löschen eines Buckets.

Verwenden der S3-Konsole

Nachdem Sie Ihren Verzeichnis-Bucket geleert und alle laufenden mehrteiligen Uploads abgebrochen haben, können Sie Ihren Bucket löschen.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie die Registerkarte Verzeichnis-Buckets.

4. Wählen Sie in der Liste Verzeichnis-Buckets die Optionsschaltfläche neben dem Bucket aus, den Sie löschen möchten.
5. Wählen Sie Löschen aus.
6. Geben Sie auf der Seite Bucket löschen den Namen des Buckets in das Textfeld ein, um das Löschen Ihres Buckets zu bestätigen.

   > **Important**
   >
   > Das Löschen eines Verzeichnis-Buckets kann nicht rückgängig gemacht werden.

7. Um Ihren Verzeichnis-Bucket zu löschen, wählen Sie Bucket löschen aus.

Verwenden der AWS-SDKs

In den folgenden Beispielen wird ein Verzeichnis-Bucket mithilfe der AWS SDK for Java 2.x und gelöschtAWS SDK for Python (Boto3).

SDK for Java 2.x

Example

```
public static void deleteBucket(S3Client s3Client, String bucketName) {

    try {
        DeleteBucketRequest del = DeleteBucketRequest.builder()
                .bucket(bucketName)
                .build();
        s3Client.deleteBucket(del);
        System.out.println("Bucket " + bucketName + " has been deleted");
    }
    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

SDK for Python

Example

```
import logging
```

```
import boto3
from botocore.exceptions import ClientError

def delete_bucket(s3_client, bucket_name):
    '''
    Delete a directory bucket in a specified Region

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket to delete; for example, 'doc-example-bucket--usw2-
az1--x-s3'
    :return: True if bucket is deleted, else False
    '''

    try:
        s3_client.delete_bucket(Bucket = bucket_name)
    except ClientError as e:
        logging.error(e)
        return False
    return True

if __name__ == '__main__':
    bucket_name = 'BUCKET_NAME'
    region = 'us-west-2'
    s3_client = boto3.client('s3', region_name = region)
```

Auflisten von Verzeichnis-Buckets

Die folgenden Beispiele zeigen, wie Verzeichnis-Buckets mithilfe der -AWSSDKs aufgelistet werden. SDKs

Verwenden der AWS-SDKs zum Auflisten von Verzeichnis-Buckets

SDK for Java 2.x

Example

Im folgenden Beispiel werden Verzeichnis-Buckets mithilfe der aufgelistetAWS SDK for Java 2.x.

```
public static void listBuckets(S3Client s3Client) {
       try {
```

```
            ListDirectoryBucketsRequest listDirectoryBucketsRequest =
 ListDirectoryBucketsRequest.builder().build();
            ListDirectoryBucketsResponse response =
 s3Client.listDirectoryBuckets(listDirectoryBucketsRequest);
            if (response.hasBuckets()) {
                for (Bucket bucket: response.buckets()) {
                    System.out.println(bucket.name());
                    System.out.println(bucket.creationDate());
                }
            }
        }

        catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Example

Im folgenden Beispiel werden Verzeichnis-Buckets mithilfe der aufgelistetAWS SDK for Python (Boto3).

```
import logging
import boto3
from botocore.exceptions import ClientError

def list_directory_buckets(s3_client):
'''
Prints a list of all directory buckets in a Region

:param s3_client: boto3 S3 client
:return: True if there are buckets in the Region, else False
'''
try:
    response = s3_client.list_directory_buckets()
    for bucket in response['Buckets']:
        print (bucket['Name'])
except ClientError as e:
    logging.error(e)
    return False
```

```
return True


if __name__ == '__main__':
    region = 'us-west-2'
    s3_client = boto3.client('s3', region_name = region)
    list_directory_buckets(s3_client)
```

AWS SDK for .NET

Example

Im folgenden Beispiel werden Verzeichnis-Buckets mithilfe der aufgelistetAWS SDK for .NET.

```
var listDirectoryBuckets = await amazonS3Client.ListDirectoryBucketsAsync(new
 ListDirectoryBucketsRequest
{
  MaxDirectoryBuckets = 10
  }).ConfigureAwait(false);
```

SDK for PHP

Example

Im folgenden Beispiel werden Verzeichnis-Buckets mithilfe der aufgelistetAWS SDK for PHP.

```
require 'vendor/autoload.php';

$s3Client = new S3Client([
    'region'      => 'us-east-1',
]);
$result = $s3Client->listDirectoryBuckets();
```

SDK for Ruby

Example

Im folgenden Beispiel werden Verzeichnis-Buckets mithilfe der aufgelistetAWS SDK for Ruby.

```
s3 = Aws::S3::Client.new(region:'us-west-2')
s3.list_directory_buckets
```

Verwenden von **`HeadBucket`** mit Verzeichnis-Buckets

Die folgenden AWS -SDK-Beispiele zeigen, wie Sie die `HeadBucket`-API-Operation verwenden, um festzustellen, ob ein Amazon S3-Verzeichnis-Bucket vorhanden ist und ob Sie über die Berechtigung für den Zugriff darauf verfügen.

Verwenden der AWS-SDKs

Das folgende AWS SDK for Java 2.x Beispiel zeigt, wie Sie feststellen, ob ein Bucket vorhanden ist und ob Sie über die Berechtigung für den Zugriff darauf verfügen.

SDK for Java 2.x

Example

AWS SDK for Java 2.x

```
public static void headBucket(S3Client s3Client, String bucketName) {
   try {
        HeadBucketRequest headBucketRequest = HeadBucketRequest
                .builder()
                .bucket(bucketName)
                .build();
        s3Client.headBucket(headBucketRequest);
        System.out.format("Amazon S3 bucket: \"%s\" found.", bucketName);
   }

   catch (S3Exception e) {
       System.err.println(e.awsErrorDetails().errorMessage());
       System.exit(1);
   }
}
```

Arbeiten mit Objekten in einem Verzeichnis-Bucket

Nachdem Sie einen Amazon S3-Verzeichnis-Bucket erstellt haben, können Sie mit Objekten arbeiten, indem Sie die Amazon S3-Konsole, AWS Command Line Interface (AWS CLI) und die -AWSSDKs verwenden. SDKs

Weitere Informationen zu Massenoperationen für Objekte mit Objekten, die in der Speicherklasse S3 Express One Zone gespeichert sind, finden Sie unter Verwaltung von Objekten. Weitere Informationen zum Importieren, Hochladen, Kopieren, Löschen und Herunterladen von Objekten und zum Lesen von Metadaten aus Objekten in Verzeichnis-Buckets finden Sie in den folgenden Themen.

Themen

Importieren von Objekten in einen Verzeichnis-Bucket

Nachdem Sie einen Verzeichnis-Bucket in Amazon S3 erstellt haben, können Sie den neuen Bucket mithilfe der Importaktion mit Daten füllen. Der Import ist eine optimierte Methode zur Erstellung von S3-Batch-Operations-Aufträgen zum Kopieren von Objekten aus Allzweck-Buckets in Verzeichnis-Buckets.

> **Note**
>
> Die folgenden Einschränkungen gelten für Importaufträge:
>
> - Der Quell-Bucket und der Ziel-Bucket müssen sich in derselben AWS-Region und demselben Konto befinden.
> - Der Quell-Bucket kann kein Verzeichnis-Bucket sein.
> - Objekte, die größer als 5 GB sind, werden nicht unterstützt und beim Kopiervorgang nicht berücksichtigt.
> - Objekte der Speicherklassen Glacier Flexible Retrieval, Glacier Deep Archive, Intelligent-Tiering Archive Access und Intelligent-Tiering Deep Archive müssen wiederhergestellt werden, bevor sie importiert werden können.

- Importierte Objekte mit MD5-Prüfsummenalgorithmen werden so konvertiert, dass sie CRC32-Prüfsummen verwenden.
- Importierte Objekte verwenden serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)
- Importierte Objekte verwenden die Speicherklasse Express One Zone, die eine andere Preisstruktur hat als die Speicherklassen, die von Allzweck-Buckets verwendet werden. Beachten Sie diesen Kostenunterschied, wenn Sie eine große Anzahl von Objekten importieren.

Wenn Sie einen Importauftrag konfigurieren, geben Sie den Quell-Bucket oder das Quellpräfix an, aus dem die vorhandenen Objekte kopiert werden. Sie stellen auch eine AWS Identity and Access Management (IAM)-Rolle bereit, die Berechtigungen für den Zugriff auf die Quellobjekte hat. Amazon S3 startet dann einen Batch-Operations-Auftrag, der die Objekte kopiert und automatisch die entsprechenden Speicherklassen- und Prüfsummeneinstellungen anwendet.

Um Importaufträge zu konfigurieren, verwenden Sie die Amazon-S3-Konsole.

Verwenden der Amazon S3-Konsole

So importieren Sie Objekte in einen Verzeichnis-Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets und dann die Registerkarte Verzeichnis-Buckets aus. Wählen Sie die Optionsschaltfläche neben dem Verzeichnis-Bucket, in den Sie Objekte importieren möchten.
3. Wählen Sie Importieren aus.
4. Geben Sie unter Quelle den Allzweck-Bucket (oder den Bucket-Pfad einschließlich Präfix) ein, der die Objekte enthält, die Sie importieren möchten. Um einen vorhandenen Allzweck-Bucket aus einer Liste auszuwählen, wählen Sie S3 durchsuchen.
5. Führen Sie für die Option Berechtigung zum Zugriff auf zum Kopieren von Quellobjekten einen der folgenden Schritte durch, um eine IAM-Rolle mit den für den Import Ihrer Quellobjekte erforderlichen Berechtigungen anzugeben:

   - Damit Amazon S3 für Sie eine neue IAM-Rolle erstellen kann, wählen Sie Neue IAM-Rolle erstellen.

> **Note**
>
> Wenn Ihre Quellobjekte mit serverseitiger Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) verschlüsselt sind, wählen Sie die Option Neue IAM-Rolle erstellen nicht aus. Geben Sie stattdessen eine vorhandene IAM-Rolle an, die über die `kms:Decrypt`-Berechtigung verfügt.
> Amazon S3 verwendet diese Berechtigung, um Ihre Objekte zu entschlüsseln. Während des Importvorgangs verschlüsselt Amazon S3 diese Objekte erneut unter Verwendung der serverseitigen Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

- Um eine bestehende IAM-Rolle aus einer Liste auszuwählen, wählen Sie Aus vorhandenen IAM-Rollen auswählen.
- Um eine bestehende IAM-Rolle anzugeben, indem Sie ihren Amazon-Ressourcennamen (ARN) angeben, wählen Sie IAM-Rollen-ARN eingeben aus und geben den ARN im entsprechenden Feld an.

6. Überprüfen Sie die Informationen, die in den Abschnitten Ziel und Kopierte Objekteinstellungen angezeigt werden. Wenn die Informationen im Bereich Ziel korrekt sind, wählen Sie Import, um den Kopierauftrag zu starten.

   Die Amazon-S3-Konsole zeigt den Status Ihres neuen Auftrags auf der Seite Batch-Operationen an. Für weitere Informationen über den Auftrag klicken Sie auf das Optionsfeld neben dem Auftragsnamen und wählen dann im Menü Aktionen die Option Details anzeigen. Um den Verzeichnis-Bucket zu öffnen, in den die Objekte importiert werden, wählen Sie Importziel anzeigen.

Verwenden von Batch Operations mit S3 Express One Zone

Sie können Amazon S3 Batch Operations verwenden, um umfangreiche Vorgänge für in S3 gespeicherte Objekte durchzuführen. Weitere Informationen zu S3 Batch Operations finden Sie unter Durchführen umfangreicher Stapelvorgänge für Amazon-S3-Objekte.

In den folgenden Themen wird das Ausführen von Batchoperationen für Objekte beschrieben, die in der Speicherklasse S3 Express One Zone in Verzeichnis-Buckets gespeichert sind.

Themen

- Verwenden von Batch Operations mit Verzeichnis-Buckets.
- Die wichtigsten Unterschiede:

Verwenden von Batch Operations mit Verzeichnis-Buckets.

Sie können den Kopiervorgang und die Invoke-AWS LambdaFunktionsvorgänge für Objekte ausführen, die in Verzeichnis-Buckets gespeichert sind. Mit Kopieren können Sie Objekte zwischen Buckets desselben Typs kopieren (z. B. aus einem Verzeichnis-Bucket in einen Verzeichnis-Bucket). Sie können auch zwischen Verzeichnis-Buckets und Allzweck-Buckets kopieren. Mit AWS Lambda-Funktion aufrufen können Sie eine Lambda-Funktion verwenden, um Aktionen für Objekte in Ihrem Verzeichnis-Bucket mit von Ihnen definiertem Code auszuführen.

Objekte kopieren

Sie können zwischen demselben Bucket-Typ oder zwischen Verzeichnis-Buckets und Allzweck-Buckets kopieren. Wenn Sie in einen Verzeichnis-Bucket kopieren, müssen Sie das richtige Format für den Amazon-Ressourcennamen (ARN) für diesen Bucket-Typ verwenden. Das ARN-Format für einen Verzeichnis-Bucket ist `arn:aws:s3express:`*`region`*`:`*`account-id`*`:bucket/`*`bucket-base-name`*`--x-s3`.

Sie können Ihren Verzeichnis-Bucket auch mit Daten füllen, indem Sie die Aktion Import in der S3-Konsole verwenden. Import ist eine optimierte Methode zur Erstellung von Batch-Operations-Aufträgen zum Kopieren von Objekten aus Allzweck-Buckets in Verzeichnis-Buckets. Für Import-Kopieraufträge aus Allzweck-Buckets in Verzeichnis-Buckets generiert S3 automatisch ein Manifest. Weitere Informationen finden Sie unter Importieren von Objekten in einen Verzeichnis-Bucket und Angeben eines Manifests.

Aufrufen von Lambda-Funktionen (**`LambdaInvoke`**)

Für die Verwendung von Batch Operations zum Aufrufen von Lambda-Funktionen, die auf Verzeichnis-Buckets wirken, gelten besondere Anforderungen. Sie müssen beispielsweise Ihre Lambda-Anforderung mithilfe eines v2 JSON-Aufrufschemas strukturieren und angeben, InvocationSchemaVersion 2.0 wann Sie den Auftrag erstellen. Weitere Informationen finden Sie unter AWS Lambda Aufrufen der Funktion .

Die wichtigsten Unterschiede:

Im Folgenden finden Sie eine Liste der wichtigsten Unterschiede bei der Verwendung von Batchoperationen zum Ausführen von Massenoperationen für Objekte, die in Verzeichnis-Buckets mit der Speicherklasse S3 Express One Zone gespeichert sind:

- Amazon S3 verschlüsselt automatisch alle neuen Objekte, die in einen S3-Bucket hochgeladen werden. Die Standardverschlüsselungskonfiguration eines S3-Buckets ist immer aktiviert und mindestens auf serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) eingestellt. Für Verzeichnis-Buckets wird nur sSSE-S3 unterstützt. Wenn Sie eine `CopyObject` Anforderung stellen, die die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) oder die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) in einem Verzeichnis-Bucket (Quelle oder Ziel) festlegt, gibt die Antwort einen HTTP-`400 (Bad Request)`Fehler zurück.
- Objekte in Verzeichnis-Buckets können nicht mit Tags markiert werden. Sie können nur einen leeren Tag-Satz angeben. Standardmäßig kopiert Batch Operations Tags. Wenn Sie ein Objekt kopieren, das Tags zwischen Allzweck-Buckets und Verzeichnis-Buckets enthält, erhalten Sie eine `-501 (Not Implemented)`Antwort.
- S3 Express One Zone bietet Ihnen die Möglichkeit, den Prüfsummenalgorithmus auszuwählen, der zur Validierung Ihrer Daten während Uploads oder Downloads verwendet wird. Sie können einen der folgenden Secure Hash Algorithms (SHA)- oder Cyclic Redundancy Check (CRC)-Algorithmen zur Überprüfung der Datenintegrität auswählen: CRC32, SHA-1 und SHA-256. MD5-based Prüfsummen werden von der Speicherklasse S3 Express One Zone nicht unterstützt.
- Standardmäßig legen alle Amazon S3-Buckets die Einstellung S3 Object Ownership auf Bucket-Eigentümer erzwungen fest und Zugriffssteuerungslisten (ACLs) sind deaktiviert. Für Verzeichnis-Buckets kann diese Einstellung nicht geändert werden. Sie können ein Objekt aus Allzweck-Buckets in Verzeichnis-Bucket kopieren. Sie können die Standard-ACL jedoch nicht überschreiben, wenn Sie in oder aus einem Verzeichnis-Bucket kopieren.
- Unabhängig davon, wie Sie Ihr Manifest angeben, muss die Liste selbst in einem Allzweck-Bucket gespeichert werden. Batch Operations kann keine vorhandenen Manifeste aus Verzeichnis-Buckets importieren (oder generierte Manifeste in speichern). Im Manifest beschriebene Objekte können jedoch in Verzeichnis-Buckets gespeichert werden.
- Batchoperationen kann keinen Verzeichnis-Bucket als Speicherort in einem S3-Inventory-Bericht angeben. Bestandsberichte unterstützen keine Verzeichnis-Buckets. Sie können eine Manifestdatei für Objekte innerhalb eines Verzeichnis-Buckets erstellen, indem Sie die `ListObjectsV2` -API-

Operation verwenden, um die Objekte aufzulisten. Anschließend können Sie die Liste in eine CSV-Datei einfügen.

Gewähren von -Zugriff

Um Kopieraufträge durchzuführen müssen Sie über die folgenden Berechtigungen verfügen:

- Um Objekte von einem Verzeichnis-Bucket zu einem anderen Verzeichnis-Bucket zu kopieren, benötigen Sie die `s3express:CreateSession`-Berechtigung.
- Um Objekte aus Verzeichnis-Buckets zu Allzweck-Buckets zu kopieren, benötigen Sie die `s3express:CreateSession`-Berechtigung und `s3:PutObject`-Berechtigung zum Schreiben der Objektkopie in den Ziel-Bucket.
- Um Objekte aus Allzweck-Buckets in Verzeichnis-Buckets zu kopieren, benötigen Sie die `-s3express:CreateSession`Berechtigung und die `-s3:GetObject`Berechtigung zum Lesen des Quellobjekts, das kopiert wird.

  Weitere Informationen finden Sie unter CopyObject in der API-Referenz zu Amazon Simple Storage Service.
- Um eine Lambda-Funktion aufzurufen, müssen Sie Ihrer Ressource basierend auf Ihrer Lambda-Funktion Berechtigungen erteilen. Um festzustellen, welche Berechtigungen erforderlich sind, überprüfen Sie die entsprechenden API-Operationen.

Hochladen eines Objekts zu einem Verzeichnis-Bucket

Nachdem Sie einen Amazon S3-Verzeichnis-Bucket erstellt haben, können Sie Objekte in diesen hochladen. Die folgenden Beispiele zeigen, wie Sie ein Objekt mithilfe der S3-Konsole und der AWS SDKs in einen Verzeichnis-Bucket hochladen. SDKs Informationen zu Massen-Upload-Operationen für Objekte mit S3 Express One Zone finden Sie unter Verwaltung von Objekten.

Verwenden der S3-Konsole

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie die Registerkarte Verzeichnis-Buckets.
4. Wählen Sie den Namen des Buckets aus, in den Sie Ihre Ordner oder Dateien hochladen möchten.

5. Wählen Sie in der Liste Objekte die Option Hochladen aus.
6. Führen Sie auf der Seite Upload einen der folgenden Schritte aus:

   - Ziehen Sie Dateien und Ordner per Drag-and-Drop in den gepunkteten Uploadbereich.
   - Wählen Sie Dateien hinzufügen oder Ordner hinzufügen, wählen Sie die hochzuladenden Dateien oder Ordner aus und klicken Sie dann auf Öffnen oder Hochladen.
7. Wählen Sie unter Prüfsummen die Prüfsummenfunktion aus, die Sie verwenden möchten.

   (Optional) Wenn Sie ein einzelnes Objekt hochladen, das kleiner als 16 MB ist, können Sie auch einen vorberechneten Prüfsummenwert angeben. Wenn Sie einen vorberechneten Wert angeben, vergleicht Amazon S3 ihn mit dem Wert, den es mithilfe der ausgewählten Prüfsummenfunktion berechnet. Wenn die Werte nicht übereinstimmen, wird der Upload nicht gestartet.
8. Die Optionen in den Abschnitten Berechtigungen und Eigenschaften werden automatisch auf Standardeinstellungen festgelegt und können nicht geändert werden. Das Blockieren des öffentlichen Zugriffs wird automatisch aktiviert, und S3-Versioning und die S3-Objektsperre können nicht für Verzeichnis-Buckets aktiviert werden.

   (Optional) Wenn Sie Ihren Objekten Metadaten in Schlüssel-Wert-Paaren hinzufügen möchten, erweitern Sie den Abschnitt Eigenschaften und wählen Sie dann im Abschnitt Metadaten die Option Metadaten hinzufügen aus.
9. Um die aufgelisteten Dateien und Ordner hochzuladen, wählen Sie Hochladen aus.

   Amazon S3 lädt Ihre Objekte und Ordner hoch. Wenn der Upload abgeschlossen ist, wird auf der Seite Upload: Status eine Erfolgsmeldung angezeigt.

Verwenden der AWS-SDKs

SDK for Java 2.x

Example

```
public static void putObject(S3Client s3Client, String bucketName, String objectKey,
 Path filePath) {
       //Using File Path to avoid loading the whole file into memory
       try {
           PutObjectRequest putObj = PutObjectRequest.builder()
                   .bucket(bucketName)
                   .key(objectKey)
```

```
                    //.metadata(metadata)
                    .build();
            s3Client.putObject(putObj, filePath);
            System.out.println("Successfully placed " + objectKey +" into bucket
 "+bucketName);

        }

        catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
}
```

SDK for Python

Example

```
import boto3
import botocore
from botocore.exceptions import ClientError


def put_object(s3_client, bucket_name, key_name, object_bytes):
    """
    Upload data to a directory bucket.
    :param s3_client: The boto3 S3 client
    :param bucket_name: The bucket that will contain the object
    :param key_name: The key of the object to be uploaded
    :param object_bytes: The data to upload
    """
    try:
        response = s3_client.put_object(Bucket=bucket_name, Key=key_name,
                            Body=object_bytes)
        print(f"Upload object '{key_name}' to bucket '{bucket_name}'.")
        return response
    except ClientError:
        print(f"Couldn't upload object '{key_name}' to bucket '{bucket_name}'.")
        raise

def main():
    # Share the client session with functions and objects to benefit from S3 Express
 One Zone auth key
    s3_client = boto3.client('s3')
```

```
    # Directory bucket name must end with --azid--x-s3
    resp = put_object(s3_client, 'doc-bucket-example--use1-az5--x-s3', 'sample.txt',
 b'Hello, World!')
    print(resp)

if __name__ == "__main__":
    main()
```

Verwenden von mehrteiligen Uploads mit Verzeichnis-Buckets

Sie können den mehrteiligen Upload-Prozess verwenden, um ein einzelnes Objekt als Satz von Teilen hochzuladen. Jeder Teil ist ein zusammenhängender Teil der Daten des Objekts. Sie können diese Objektteile unabhängig und in beliebiger Reihenfolge hochladen. Wenn die Übertragung eines Teils fehlschlägt, können Sie das Teil erneut übertragen, ohne dass dies Auswirkungen auf andere Teile hat. Nachdem alle Teile Ihres Objekts hochgeladen sind, fügt Amazon S3 diese Teile zusammen und erstellt das Objekt. Wenn Ihre Objektgröße 100 MB erreicht, sollten Sie in der Regel mehrteilige Uploads verwenden, anstatt das Objekt in einem einzigen Vorgang hochzuladen.

Die Nutzung mehrteiliger Uploads bietet die folgenden Vorteile:

- Verbesserter Durchsatz – Sie können die Teile parallel hochladen, um den Durchsatz zu erhöhen.
- Schnelle Wiederherstellung nach Netzwerkproblemen – Kleinere Teilegrößen minimieren die Auswirkungen eines Neustarts eines fehlgeschlagenen Uploads aufgrund eines Netzwerkfehlers.
- Anhalten und Fortsetzen von Objekt-Uploads – Sie können Objektteile mit der Zeit hochladen. Nachdem Sie einen mehrteiligen Upload initiiert haben, gibt es kein Ablaufdatum. Sie müssen den mehrteiligen Upload explizit abschließen oder abbrechen.
- Starten Sie einen Upload, bevor Sie die endgültige Objektgröße kennen – Sie können ein Objekt hochladen, während Sie es noch erstellen.

Wir empfehlen Ihnen, mehrteilige Uploads wie folgt zu verwenden:

- Wenn Sie große Objekte über ein stabiles Netzwerk mit hoher Bandbreite hochladen, verwenden Sie mehrteilige Uploads, um die Nutzung Ihrer verfügbaren Bandbreite zu maximieren, indem Sie Objektteile parallel hochladen, um eine Multi-Thread-Leistung zu erzielen.
- Wenn Sie über ein fehlerhaftes Netzwerk hochladen, verwenden Sie mehrteilige Uploads, um die Ausfallsicherheit bei Netzwerkfehlern zu erhöhen, indem Sie Upload-Neustarts vermeiden. Wenn Sie mehrteilige Uploads verwenden, müssen Sie nur die Teile erneut hochladen, die während

des Uploads unterbrochen werden. Sie müssen nicht das gesamte Objekt von Anfang an neu hochladen.

Wenn Sie mehrteilige Uploads verwenden, um Objekte in die Speicherklasse Amazon S3 Express One Zone in Verzeichnis-Buckets hochzuladen, ähnelt der Prozess des mehrteiligen Uploads, um Objekte in Allzweck-Buckets hochzuladen. Es gibt jedoch einige grundlegende Unterschiede.

Weitere Informationen zur Verwendung von mehrteiligen Uploads zum Hochladen von Objekten in S3 Express One Zone finden Sie in den folgenden Themen.

Themen

Der mehrteilige Upload-Prozess

Ein mehrteiliger Upload ist ein dreistufiger Prozess:

- Sie initiieren den Upload.
- Sie laden die Objektteile hoch.
- Nachdem Sie alle Teile hochgeladen haben, schließen Sie den mehrteiligen Upload ab.

Nach Erhalt der Anforderung zum Abschluss des mehrteiligen Uploads erstellt Amazon S3 das Objekt aus den hochgeladenen Teilen, und Sie können dann wie jedes andere Objekt in Ihrem Bucket auf das Objekt zugreifen.

Initiieren des mehrteiligen Uploads

Wenn Sie eine Anfrage zum Initiieren eines mehrteiligen Uploads senden, gibt Amazon S3 eine Antwort mit einer Upload-ID zurück, als eindeutige Kennung für den Multipart-Upload. Sie müssen diese Upload-ID immer angeben, wenn Sie Teile hochladen, die Teile auflisten, einen Upload durchführen oder ihn abbrechen.

Teile hochladen

Beim Hochladen eines Teils müssen Sie zusätzlich zur Upload-ID eine Teilenummer angeben. Wenn Sie einen mehrteiligen Upload mit S3 Express One Zone verwenden, müssen die mehrteiligen Teilenummern aufeinander folgen. Wenn Sie versuchen, eine mehrteilige Upload-Anforderung mit nicht aufeinanderfolgenden Teilenummern abzuschließen, wird ein Fehler `HTTP 400 Bad Request` (Ungültige Teilebestellung) generiert.

Eine Teilenummer identifiziert einen Teil und seine Position in dem Objekt, das Sie hochladen, eindeutig. Wenn Sie einen neuen Teil mit derselben Teilenummer wie ein zuvor hochgeladener Teil hochladen, wird der zuvor hochgeladene Teil überschrieben.

Wenn Sie einen Teil hochladen, gibt Amazon S3 einen entity tag (ETag)-Header in der Antwort zurück. Für jeden Teilupload müssen Sie die Teilenummer und den ETag-Wert notieren. Die ETag-Werte für alle Uploads von Objektteilen bleiben gleich, aber jedem Teil wird eine andere Teilenummer zugewiesen. Sie müssen diese Werte in die spätere Anforderung einschließen, um den mehrteiligen Upload abzuschließen.

Amazon S3 verschlüsselt automatisch alle neuen Objekte, die in einen S3-Bucket hochgeladen werden. Wenn Sie bei einem mehrteiligen Upload in Ihrer Kopieranforderung keine Verschlüsselungsinformationen angeben, wird die Verschlüsselungseinstellung der hochgeladenen Teile auf die Standardverschlüsselungskonfiguration des Ziel-Buckets festgelegt. Die Standardverschlüsselungskonfiguration eines Amazon-S3-Buckets ist immer aktiviert und mindestens auf serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) eingestellt. Für Verzeichnis-Buckets wird nur SSE-S3 unterstützt. Weitere Informationen finden Sie unter Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

Abschließen eines mehrteiligen Uploads

Wenn Sie einen mehrteiligen Upload abschließen, erstellt Amazon S3 das Objekt, indem die Teile in aufsteigender Reihenfolge auf der Grundlage der Teilenummer verkettet werden. Nach einer erfolgreich ausgeführten Abschlussanforderung sind die Teile nicht mehr vorhanden.

Ihre Anforderung zum Abschluss eines mehrteiligen Uploads muss die Upload-ID und eine Liste der Teilenummern und der entsprechenden ETag-Werte enthalten. Die Amazon-S3-Antwort enthält einen ETag, der die kombinierten Objektdaten eindeutig identifiziert. Dieses ETag ist kein MD5-Hash der Objektdaten.

Auflistungen mehrteiliger Uploads

Sie können alle Teile eines bestimmten Multipart-Uploads oder alle laufenden mehrteiligen Uploads auflisten. Die Operation für die Teileauflistung gibt die Teileinformationen zurück, die Sie für einen bestimmten mehrteiligen Upload hochgeladen haben. Für jeden Abruf einer Teileauflistung gibt Amazon S3 die Teileinformationen für einen angegebenen mehrteiligen Upload bis zu maximal 1.000 Teilen zurück. Wenn in dem mehrteiligen Upload mehr als 1 000 Teile vorhanden sind, müssen Sie die Paginierung verwenden, um alle Teile abzurufen.

Die zurückgegebene Teileliste enthält keine Teile, deren Upload noch nicht abgeschlossen ist. Bei Verwendung der Operation Mehrteilige Uploads auflisten können Sie eine Liste aller mehrteiligen Uploads erhalten, die sich in Bearbeitung befinden.

Ein laufender mehrteiliger Upload ist ein Upload, den Sie initiiert haben, der aber noch nicht abgeschlossen ist oder abgebrochen wurde. Jeder Anforderung gibt bis zu 1.000 mehrteilige Uploads zurück. Wenn mehr als 1 000 mehrteilige Uploads vorhanden sind, müssen Sie zusätzliche Anforderungen senden, um die verbleibenden mehrteiligen Uploads abzurufen. Verwenden Sie die zurückgegebene Liste nur zur Überprüfung. Verwenden Sie das Ergebnis dieser Auflistung nicht, wenn Sie eine Anforderung für den Abschluss eines mehrteiligen Uploads senden. Halten Sie sich stattdessen an Ihre eigene Liste der Teilenummern, die Sie beim Hochladen von Teilen angegeben haben, und die diesbezüglichen ETag-Werte, die Amazon S3 zurückgegeben hat.

Weitere Informationen zu mehrteiligen Upload-Auflistungen finden Sie unter ListParts in der API-Referenz zu Amazon Simple Storage Service.

Prüfsummen mit mehrteiligen Upload-Operationen

Wenn Sie ein Objekt hochladen, können Sie einen Prüfsummenalgorithmus angeben, um die Objektintegrität zu überprüfen. MD5 wird für Verzeichnis-Buckets nicht unterstützt. Sie können einen der folgenden Secure Hash Algorithms (SHA)- oder Zyklic Redundancy Check (CRC)-Algorithmen für die Datenintegritätsprüfung angeben:

- CRC32
- CRC32C
- SHA-1
- SHA-256

Sie können die Amazon S3-REST-API oder die AWS -SDKs verwenden, um den Prüfsummenwert für einzelne Teile mithilfe von `GetObject` oder abzurufen`HeadObject`. Wenn Sie die

Prüfsummenwerte für einzelne Teile von mehrteiligen Uploads abrufen möchten, die noch in Bearbeitung sind, können Sie `ListParts` verwenden.

> **Important**
>
> Bei Verwendung der vorherigen Prüfsummenalgorithmen müssen die mehrteiligen Teilenummern aufeinanderfolgende Teilenummern verwenden. Wenn Sie versuchen, eine mehrteilige Upload-Anforderung mit nicht aufeinanderfolgenden Teilenummern abzuschließen, generiert Amazon S3 einen Fehler `HTTP 400 Bad Request` (Ungültige Teilebestellung).

Weitere Informationen zur Funktionsweise von Prüfsummen mit mehrteiligen Objekten finden Sie unterÜberprüfung der Objektintegrität.

Gleichzeitige mehrteilige Upload-Vorgänge

In einer verteilten Entwicklungsumgebung kann Ihre Anwendung mehrere Aktualisierungen für dasselbe Objekt gleichzeitig initiieren. Ihre Anwendung kann beispielsweise mehrere mehrteilige Uploads initiieren, indem sie denselben Objektschlüssel verwendet. Für jeden dieser Uploads kann Ihre Anwendung Teile hochladen und eine Anfrage auf Abschluss des Uploads an Amazon S3 senden, um das Objekt zu erstellen. Für S3 Express One Zone entspricht die Objekterstellungszeit dem Abschlussdatum des mehrteiligen Uploads.

> **Note**
>
> Zwischen dem Zeitpunkt, an dem Sie einen mehrteiligen Upload initiieren und diesen dann abschließen, ist es möglich, dass eine weitere Anfrage, die von Amazon S3 empfangen wird, Vorrang hat. Angenommen, Sie initiieren einen mehrteiligen Upload mit dem Schlüsselnamen `largevideo.mp4`. Bevor Sie den Upload abschließen, löscht eine andere Operation den `largevideo.mp4` Schlüssel. In diesem Fall kann die Antwort auf den vollständigen mehrteiligen Upload auf eine erfolgreiche Objekterstellung für hinweisen, `largevideo.mp4` ohne dass Sie das Objekt jemals sehen.

> **Important**
>
> Versioning wird für Objekte, die in Verzeichnis-Buckets gespeichert sind, nicht unterstützt.

Mehrteilige Uploads und Preise

Nachdem Sie einen mehrteiligen Upload gestartet haben, behält Amazon S3 alle Teile bei, bis Sie den Upload abschließen oder abbrechen. Während seiner gesamten Lebensdauer werden Ihnen der gesamte Speicher, die Bandbreite und die Anforderungen für diesen mehrteiligen Upload und die zugehörigen Teile in Rechnung gestellt. Wenn Sie den mehrteiligen Upload abbrechen, löscht Amazon S3 die Upload-Artefakte und alle Teile, die Sie hochgeladen haben, und Ihnen werden diese nicht mehr in Rechnung gestellt. Für das Löschen unvollständiger mehrteiliger Uploads fallen keine Gebühren für vorzeitiges Löschen an, unabhängig von der angegebenen Speicherklasse. Weitere Informationen zu Preisen finden Sie unter Amazon-S3-Preise.

> **Important**
>
> Wenn die Anforderung zum Abschluss des mehrteiligen Uploads nicht erfolgreich gesendet wurde, werden die Objektteile zusammengestellt und ein Objekt wird erstellt. Ihnen wird der gesamte Speicher in Rechnung gestellt, der mit hochgeladenen Teilen verknüpft ist. Es ist wichtig, dass Sie entweder den mehrteiligen Upload abschließen, damit das Objekt erstellt wird, oder den mehrteiligen Upload abbrechen, um alle hochgeladenen Teile zu entfernen. Bevor Sie einen Verzeichnis-Bucket löschen können, müssen Sie alle laufenden mehrteiligen Uploads abschließen oder abbrechen. Verzeichnis-Buckets unterstützen keine S3-Lebenszykluskonfigurationen. Bei Bedarf können Sie Ihre aktiven mehrteiligen Uploads auflisten, dann die Uploads abbrechen und dann Ihren Bucket löschen.

API-Operationen und Berechtigungen für mehrteilige Uploads

Um den Zugriff auf Objektverwaltungs-API-Operationen in einem Verzeichnis-Bucket zu erlauben, erteilen Sie die `-s3express:CreateSession`Berechtigung in einer Bucket-Richtlinie oder einer AWS Identity and Access Management (IAM) identitätsbasierten Richtlinie.

Sie müssen über die erforderlichen Berechtigungen verfügen, um die Multipart-Upload-Vorgänge zu verwenden. Sie können Bucket-Richtlinien oder identitätsbasierte IAM-Richtlinien verwenden, um IAM-Prinzipalen Berechtigungen zum Ausführen dieser Operationen zu erteilen. Die folgende Tabelle listet die erforderlichen Berechtigungen für verschiedene mehrteilige Uploadvorgänge auf.

Sie können den Initiator eines mehrteiligen Uploads über das `-Initiator`Element identifizieren. Wenn der Initiator ein istAWS-Konto, stellt dieses Element die gleichen Informationen wie das `0wner` Element bereit. Wenn der Initiator ein IAM-Benutzer ist, stellt dieses Element den Benutzer-ARN und den Anzeigenamen bereit.

Aktion	Erforderliche Berechtigungen
Erstellen eines mehrteiligen Uploads	Um den mehrteiligen Upload zu erstellen, müssen Sie die `s3express:CreateSession` Aktion für den Verzeichnis-Bucket ausführen können.
Initiieren eines mehrteiligen Uploads	Um den mehrteiligen Upload zu initiieren, müssen Sie die `s3express:CreateSession` Aktion für den Verzeichnis-Bucket ausführen können.
Hochladen eines Teils	Um einen Teil hochzuladen, müssen Sie die `s3express:CreateSession` Aktion für den Verzeichnis-Bucket ausführen können. Damit der Initiator einen Teil hochladen kann, muss der Bucket-Eigentümer dem Initiator erlauben, die `s3express:CreateSession` Aktion für den Verzeichnis-Bucket auszuführen.
Hochladen eines Teils (Kopieren)	Um einen Teil hochzuladen, müssen Sie die `s3express:CreateSession` Aktion für den Verzeichnis-Bucket ausführen können. Damit der Initiator einen Teil eines Objekts hochladen kann, muss der Bucket-Besitzer dem Initiator eine Genehmigung für die Ausführung der Aktion `s3express:CreateSession` für das Objekt erteilen.
Abschließen eines mehrteiligen Uploads	Um einen mehrteiligen Upload abzuschließen, müssen Sie die `s3express:CreateSession` Aktion für den Verzeichnis-Bucket ausführen können. Damit der Initiator einen mehrteiligen Upload abschließen kann, muss der Bucket-Eigentümer dem Initiator erlauben, die `s3express:CreateSession` Aktion für das Objekt auszuführen.
Abbrechen eines mehrteiligen Uploads	Um einen mehrteiligen Upload abzubrechen, müssen Sie die `s3express:CreateSession` Aktion ausführen können. Damit der Initiator einen mehrteiligen Upload abbrechen kann, muss dem Initiator explizite Zugriffserlaubnis gewährt werden, um die `s3express:CreateSession` Aktion ausführen zu können.

Aktion	Erforderliche Berechtigungen
Auflisten von Teilen	Um die Teile in einem mehrteiligen Upload aufzulisten, müssen Sie die `s3express:CreateSession` Aktion für den Verzeichnis-Bucket ausführen können.
Laufende mehrteilige Uploads auflisten	Um die laufenden mehrteiligen Uploads in einen Bucket aufzulisten, müssen Sie die `s3:ListBucketMultipartUploads` Aktion für diesen Bucket ausführen können.

API-Operationsunterstützung für mehrteilige Uploads

In den folgenden Abschnitten der Amazon Simple Storage Service API Reference werden die Amazon S3 REST API-Operationen für mehrteilige Uploads beschrieben.

- CreateMultipartUpload
- UploadPart
- UploadPartCopy
- CompleteMultipartUpload
- AbortMultipartUpload
- ListParts
- ListMultipartUploads

Beispiele

Informationen zur Verwendung eines mehrteiligen Uploads zum Hochladen eines Objekts in S3 Express One Zone in einem Verzeichnis-Bucket finden Sie in den folgenden Beispielen.

Themen

- Erstellen eines mehrteiligen Uploads
- Hochladen der Teile eines mehrteiligen Uploads
- Abschließen eines mehrteiligen Uploads
- Abbrechen eines mehrteiligen Uploads
- Erstellen eines mehrteiligen Upload-Kopiervorgangs

- Auflisten von laufenden mehrteiligen Uploads
- Auflisten der Teile eines mehrteiligen Uploads

Erstellen eines mehrteiligen Uploads

Die folgenden Beispiele zeigen, wie Sie einen mehrteiligen Upload mithilfe der AWS SDK for Java 2.x und erstellenAWS SDK for Python (Boto3).

SDK for Java 2.x

Example

```
/**
 * This method creates a multipart upload request that generates a unique upload ID
 that is used to track
 * all the upload parts
 *
 * @param s3
 * @param bucketName - for example, 'doc-example-bucket--use1-az4--x-s3'
 * @param key
 * @return
 */
 private static String createMultipartUpload(S3Client s3, String bucketName, String
 key) {

     CreateMultipartUploadRequest createMultipartUploadRequest =
 CreateMultipartUploadRequest.builder()
             .bucket(bucketName)
             .key(key)
             .build();

     String uploadId = null;

     try {
         CreateMultipartUploadResponse response =
 s3.createMultipartUpload(createMultipartUploadRequest);
         uploadId = response.uploadId();
     }
     catch (S3Exception e) {
         System.err.println(e.awsErrorDetails().errorMessage());
         System.exit(1);
     }
```

```
     return uploadId;
```

SDK for Python

Example

```
def create_multipart_upload(s3_client, bucket_name, key_name):
    '''
   Create a multipart upload to a directory bucket

   :param s3_client: boto3 S3 client
   :param bucket_name: The destination bucket for the multipart upload
   :param key_name: The key name for the object to be uploaded
   :return: The UploadId for the multipart upload if created successfully, else None
   '''

   try:
        mpu = s3_client.create_multipart_upload(Bucket = bucket_name, Key =
 key_name)
        return mpu['UploadId']
    except ClientError as e:
        logging.error(e)
        return None
```

Hochladen der Teile eines mehrteiligen Uploads

Die folgenden Beispiele zeigen, wie Sie ein einzelnes Objekt in Teile aufteilen und diese Teile dann mithilfe des SDK for Java 2.x und SDK for Python in einen Verzeichnis-Bucket hochladen.

SDK for Java 2.x

Example

```
/**
 * This method creates part requests and uploads individual parts to S3 and then
 returns all the completed parts
 *
 * @param s3
 * @param bucketName
 * @param key
 * @param uploadId
 * @throws IOException
```

```
*/
private static ListCompletedPartmultipartUpload(S3Client s3, String bucketName,
String key, String uploadId, String filePath) throws IOException {

        int partNumber = 1;
        ListCompletedPart completedParts = new ArrayList<>();
        ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

        // read the local file, breakdown into chunks and process
        try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
            long fileSize = file.length();
            int position = 0;
            while (position < fileSize) {
                file.seek(position);
                int read = file.getChannel().read(bb);

                bb.flip(); // Swap position and limit before reading from the
buffer.
                UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                        .bucket(bucketName)
                        .key(key)
                        .uploadId(uploadId)
                        .partNumber(partNumber)
                        .build();

                UploadPartResponse partResponse = s3.uploadPart(
                        uploadPartRequest,
                        RequestBody.fromByteBuffer(bb));

                CompletedPart part = CompletedPart.builder()
                        .partNumber(partNumber)
                        .eTag(partResponse.eTag())
                        .build();
                completedParts.add(part);

                bb.clear();
                position += read;
                partNumber++;
            }
        }

        catch (IOException e) {
            throw e;
        }
```

```
        return completedParts;
    }
```

SDK for Python

Example

```
def multipart_upload(s3_client, bucket_name, key_name, mpu_id, part_size):
    '''
    Break up a file into multiple parts and upload those parts to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Destination bucket for the multipart upload
    :param key_name: Key name for object to be uploaded and for the local file
 that's being uploaded
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_size: The size parts that the object will be broken into, in bytes.
                      Minimum 5 MiB, Maximum 5 GiB. There is no minimum size for the
 last part of your multipart upload.
    :return: part_list for the multipart upload if all parts are uploaded
 successfully, else None
    '''

    part_list = []
    try:
        with open(key_name, 'rb') as file:
            part_counter = 1
            while True:
                file_part = file.read(part_size)
                if not len(file_part):
                    break
                upload_part = s3_client.upload_part(
                    Bucket = bucket_name,
                    Key = key_name,
                    UploadId = mpu_id,
                    Body = file_part,
                    PartNumber = part_counter
                )
                part_list.append({'PartNumber': part_counter, 'ETag':
 upload_part['ETag']})
                part_counter += 1
    except ClientError as e:
        logging.error(e)
        return None
```

```
    return part_list
```

Abschließen eines mehrteiligen Uploads

Die folgenden Beispiele zeigen, wie Sie einen mehrteiligen Upload mit dem SDK for Java 2.x und dem SDK for Python abschließen.

SDK for Java 2.x

Example

```
/**
 * This method completes the multipart upload request by collating all the upload
 parts
 * @param s3
 * @param bucketName - for example, 'doc-example-bucket--usw2-az1--x-s3'
 * @param key
 * @param uploadId
 * @param uploadParts
 */
 private static void completeMultipartUpload(S3Client s3, String bucketName, String
 key, String uploadId, ListCompletedPart uploadParts) {
        CompletedMultipartUpload completedMultipartUpload =
 CompletedMultipartUpload.builder()
                .parts(uploadParts)
                .build();

        CompleteMultipartUploadRequest completeMultipartUploadRequest =
                CompleteMultipartUploadRequest.builder()
                        .bucket(bucketName)
                        .key(key)
                        .uploadId(uploadId)
                        .multipartUpload(completedMultipartUpload)
                        .build();

        s3.completeMultipartUpload(completeMultipartUploadRequest);
    }

    public static void multipartUploadTest(S3Client s3, String bucketName, String
 key, String localFilePath)  {
        System.out.println("Starting multipart upload for: " + key);
        try {
            String uploadId = createMultipartUpload(s3, bucketName, key);
```

```
            System.out.println(uploadId);
            ListCompletedPart parts = multipartUpload(s3, bucketName, key, uploadId,
 localFilePath);
            completeMultipartUpload(s3, bucketName, key, uploadId, parts);
            System.out.println("Multipart upload completed for: " + key);
        }

        catch (Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Example

```
def complete_multipart_upload(s3_client, bucket_name, key_name, mpu_id, part_list):
    '''
    Completes a multipart upload to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: The destination bucket for the multipart upload
    :param key_name: The key name for the object to be uploaded
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_list: The list of uploaded part numbers with their associated ETags
    :return: True if the multipart upload was completed successfully, else False
    '''

    try:
        s3_client.complete_multipart_upload(
            Bucket = bucket_name,
            Key = key_name,
            UploadId = mpu_id,
            MultipartUpload = {
                'Parts': part_list
            }
        )
    except ClientError as e:
        logging.error(e)
        return False
    return True

if __name__ == '__main__':
```

```
    MB = 1024 ** 2
    region = 'us-west-2'
    bucket_name = 'BUCKET_NAME'
    key_name = 'OBJECT_NAME'
    part_size = 10 * MB
    s3_client = boto3.client('s3', region_name = region)
    mpu_id = create_multipart_upload(s3_client, bucket_name, key_name)
    if mpu_id is not None:
        part_list = multipart_upload(s3_client, bucket_name, key_name, mpu_id,
part_size)
        if part_list is not None:
            if complete_multipart_upload(s3_client, bucket_name, key_name, mpu_id,
part_list):
                print (f'{key_name} successfully uploaded through a ultipart upload
to {bucket_name}')
            else:
                print (f'Could not upload {key_name} hrough a multipart upload to
{bucket_name}')
```

Abbrechen eines mehrteiligen Uploads

Die folgenden Beispiele zeigen, wie Sie einen mehrteiligen Upload mit dem SDK for Java 2.x und dem SDK for Python abbrechen.

SDK for Java 2.x

Example

```
public static void abortMultiPartUploads( S3Client s3, String bucketName ) {

        try {
            ListMultipartUploadsRequest listMultipartUploadsRequest =
 ListMultipartUploadsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListMultipartUploadsResponse response =
 s3.listMultipartUploads(listMultipartUploadsRequest);
            ListMultipartUpload uploads = response.uploads();

            AbortMultipartUploadRequest abortMultipartUploadRequest;
            for (MultipartUpload upload: uploads) {
                abortMultipartUploadRequest = AbortMultipartUploadRequest.builder()
```

```
                        .bucket(bucketName)
                        .key(upload.key())
                        .uploadId(upload.uploadId())
                        .build();

                s3.abortMultipartUpload(abortMultipartUploadRequest);
            }

        }

        catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Example

```
import logging
import boto3
from botocore.exceptions import ClientError


def abort_multipart_upload(s3_client, bucket_name, key_name, upload_id):
    '''
    Aborts a partial multipart upload in a directory bucket.

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket where the multipart upload was initiated - for
 example, 'doc-example-bucket--usw2-az1--x-s3'
    :param key_name: Name of the object for which the multipart upload needs to be
 aborted
    :param upload_id: Multipart upload ID for the multipart upload to be aborted
    :return: True if the multipart upload was successfully aborted, False if not
    '''
    try:
        s3_client.abort_multipart_upload(
            Bucket = bucket_name,
            Key = key_name,
            UploadId = upload_id
        )
    except ClientError as e:
```

```
        logging.error(e)
        return False
    return True


if __name__ == '__main__':
    region = 'us-west-2'
    bucket_name = 'BUCKET_NAME'
    key_name = 'KEY_NAME'
        upload_id = 'UPLOAD_ID'
    s3_client = boto3.client('s3', region_name = region)
    if abort_multipart_upload(s3_client, bucket_name, key_name, upload_id):
        print (f'Multipart upload for object {key_name} in {bucket_name} bucket has
 been aborted')
    else:
        print (f'Unable to abort multipart upload for object {key_name} in
 {bucket_name} bucket')
```

Erstellen eines mehrteiligen Upload-Kopiervorgangs

Die folgenden Beispiele zeigen, wie Sie einen mehrteiligen Upload verwenden, um ein Objekt mithilfe von SDK for Java 2.x und SDK for Python programmgesteuert von einem Bucket in einen anderen zu kopieren.

SDK for Java 2.x

Example

```
/**
 * This method creates a multipart upload request that generates a unique upload ID
 that is used to track
 * all the upload parts.
 *
 * @param s3
 * @param bucketName
 * @param key
 * @return
 */
 private static String createMultipartUpload(S3Client s3, String bucketName, String
 key) {
        CreateMultipartUploadRequest createMultipartUploadRequest =
 CreateMultipartUploadRequest.builder()
```

```
                .bucket(bucketName)
                .key(key)
                .build();
        String uploadId = null;
        try {
            CreateMultipartUploadResponse response =
s3.createMultipartUpload(createMultipartUploadRequest);
            uploadId = response.uploadId();
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return uploadId;
 }

 /**
  * Creates copy parts based on source object size and copies over individual parts
  *
  * @param s3
  * @param sourceBucket
  * @param sourceKey
  * @param destnBucket
  * @param destnKey
  * @param uploadId
  * @return
  * @throws IOException
  */
   public static ListCompletedPart multipartUploadCopy(S3Client s3, String
sourceBucket, String sourceKey, String destnBucket, String destnKey, String
uploadId) throws IOException {

        // Get the object size to track the end of the copy operation.
        HeadObjectRequest headObjectRequest = HeadObjectRequest
                .builder()
                .bucket(sourceBucket)
                .key(sourceKey)
                .build();
        HeadObjectResponse response = s3.headObject(headObjectRequest);
        Long objectSize = response.contentLength();

        System.out.println("Source Object size: " + objectSize);

        // Copy the object using 20 MB parts.
        long partSize = 20 * 1024 * 1024;
```

```
        long bytePosition = 0;
        int partNum = 1;
        ListCompletedPart completedParts = new ArrayList<>();
        while (bytePosition < objectSize) {
            // The last part might be smaller than partSize, so check to make sure
            // that lastByte isn't beyond the end of the object.
            long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

            System.out.println("part no: " + partNum + ", bytePosition: " +
bytePosition + ", lastByte: " + lastByte);

            // Copy this part.
            UploadPartCopyRequest req = UploadPartCopyRequest.builder()
                    .uploadId(uploadId)
                    .sourceBucket(sourceBucket)
                    .sourceKey(sourceKey)
                    .destinationBucket(destnBucket)
                    .destinationKey(destnKey)
                    .copySourceRange("bytes="+bytePosition+"-"+lastByte)
                    .partNumber(partNum)
                    .build();
            UploadPartCopyResponse res = s3.uploadPartCopy(req);
            CompletedPart part = CompletedPart.builder()
                    .partNumber(partNum)
                    .eTag(res.copyPartResult().eTag())
                    .build();
            completedParts.add(part);
            partNum++;
            bytePosition += partSize;
        }
        return completedParts;
    }


    public static void multipartCopyUploadTest(S3Client s3, String srcBucket, String
srcKey, String destnBucket, String destnKey)  {
        System.out.println("Starting multipart copy for: " + srcKey);
        try {
            String uploadId = createMultipartUpload(s3, destnBucket, destnKey);
            System.out.println(uploadId);
            ListCompletedPart parts = multipartUploadCopy(s3, srcBucket,
srcKey,destnBucket,  destnKey, uploadId);
            completeMultipartUpload(s3, destnBucket, destnKey, uploadId, parts);
            System.out.println("Multipart copy completed for: " + srcKey);
```

```
        } catch (Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Example

```
import logging
import boto3
from botocore.exceptions import ClientError

def head_object(s3_client, bucket_name, key_name):
    '''
    Returns metadata for an object in a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket that contains the object to query for metadata
    :param key_name: Key name to query for metadata
    :return: Metadata for the specified object if successful, else None
    '''

    try:
        response = s3_client.head_object(
            Bucket = bucket_name,
            Key = key_name
        )
        return response
    except ClientError as e:
        logging.error(e)
        return None

def create_multipart_upload(s3_client, bucket_name, key_name):
    '''
    Create a multipart upload to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Destination bucket for the multipart upload
    :param key_name: Key name of the object to be uploaded
    :return: UploadId for the multipart upload if created successfully, else None
    '''
```

```
    try:
        mpu = s3_client.create_multipart_upload(Bucket = bucket_name, Key =
 key_name)
        return mpu['UploadId']
    except ClientError as e:
        logging.error(e)
        return None

def multipart_copy_upload(s3_client, source_bucket_name, key_name,
 target_bucket_name, mpu_id, part_size):
    '''
    Copy an object in a directory bucket to another bucket in multiple parts of a
 specified size

    :param s3_client: boto3 S3 client
    :param source_bucket_name: Bucket where the source object exists
    :param key_name: Key name of the object to be copied
    :param target_bucket_name: Destination bucket for copied object
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_size: The size parts that the object will be broken into, in bytes.
                      Minimum 5 MiB, Maximum 5 GiB. There is no minimum size for the
 last part of your multipart upload.
    :return: part_list for the multipart copy if all parts are copied successfully,
 else None
    '''

    part_list = []
    copy_source = {
        'Bucket': source_bucket_name,
        'Key': key_name
    }
    try:
        part_counter = 1
        object_size = head_object(s3_client, source_bucket_name, key_name)
        if object_size is not None:
            object_size = object_size['ContentLength']
        while (part_counter - 1) * part_size <object_size:
            bytes_start = (part_counter - 1) * part_size
            bytes_end = (part_counter * part_size) - 1
            upload_copy_part = s3_client.upload_part_copy (
                Bucket = target_bucket_name,
                CopySource = copy_source,
                CopySourceRange = f'bytes={bytes_start}-{bytes_end}',
                Key = key_name,
```

```
                PartNumber = part_counter,
                UploadId = mpu_id
            )
            part_list.append({'PartNumber': part_counter, 'ETag':
 upload_copy_part['CopyPartResult']['ETag']})
            part_counter += 1
    except ClientError as e:
        logging.error(e)
        return None
    return part_list

def complete_multipart_upload(s3_client, bucket_name, key_name, mpu_id, part_list):
    '''
    Completes a multipart upload to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Destination bucket for the multipart upload
    :param key_name: Key name of the object to be uploaded
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_list: List of uploaded part numbers with associated ETags
    :return: True if the multipart upload was completed successfully, else False
    '''

    try:
        s3_client.complete_multipart_upload(
            Bucket = bucket_name,
            Key = key_name,
            UploadId = mpu_id,
            MultipartUpload = {
                'Parts': part_list
            }
        )
    except ClientError as e:
        logging.error(e)
        return False
    return True

if __name__ == '__main__':
    MB = 1024 ** 2
    region = 'us-west-2'
    source_bucket_name = 'SOURCE_BUCKET_NAME'
    target_bucket_name = 'TARGET_BUCKET_NAME'
    key_name = 'KEY_NAME'
    part_size = 10 * MB
```

```
   s3_client = boto3.client('s3', region_name = region)
   mpu_id = create_multipart_upload(s3_client, target_bucket_name, key_name)
   if mpu_id is not None:
       part_list = multipart_copy_upload(s3_client, source_bucket_name, key_name,
target_bucket_name, mpu_id, part_size)
       if part_list is not None:
           if complete_multipart_upload(s3_client, target_bucket_name, key_name,
mpu_id, part_list):
               print (f'{key_name} successfully copied through multipart copy from
{source_bucket_name} to {target_bucket_name}')
           else:
               print (f'Could not copy {key_name} through multipart copy from
{source_bucket_name} to {target_bucket_name}')
```

Auflisten von laufenden mehrteiligen Uploads

Die folgenden Beispiele zeigen, wie Sie laufende (unvollständige) mehrteilige Uploads mit dem SDK for Java 2.x und dem SDK for Python auflisten.

SDK for Java 2.x

Example

```
public static void listMultiPartUploads( S3Client s3, String bucketName) {
       try {
           ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
               .bucket(bucketName)
               .build();

           ListMultipartUploadsResponse response =
s3.listMultipartUploads(listMultipartUploadsRequest);
           List MultipartUpload uploads = response.uploads();
           for (MultipartUpload upload: uploads) {
               System.out.println("Upload in progress: Key = \"" + upload.key() +
"\", id = " + upload.uploadId());
           }
     }
     catch (S3Exception e) {
           System.err.println(e.getMessage());
           System.exit(1);
     }
```

```
  }
```

SDK for Python

Example

```
import logging
import boto3
from botocore.exceptions import ClientError

def list_multipart_uploads(s3_client, bucket_name):
    '''
    List any incomplete multipart uploads in a directory bucket in e specified gion

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket to check for incomplete multipart uploads
    :return: List of incomplete multipart uploads if there are any, None if not
    '''

    try:
        response = s3_client.list_multipart_uploads(Bucket = bucket_name)
        if 'Uploads' in response.keys():
            return response['Uploads']
        else:
            return None
    except ClientError as e:
        logging.error(e)

if __name__ == '__main__':
    bucket_name = 'BUCKET_NAME'
    region = 'us-west-2'
    s3_client = boto3.client('s3', region_name = region)
    multipart_uploads = list_multipart_uploads(s3_client, bucket_name)
    if multipart_uploads is not None:
        print (f'There are {len(multipart_uploads)} ncomplete multipart uploads for
 {bucket_name}')
    else:
        print (f'There are no incomplete multipart uploads for {bucket_name}')
```

Auflisten der Teile eines mehrteiligen Uploads

Die folgenden Beispiele zeigen, wie Sie die Teile eines mehrteiligen Uploads mit dem SDK for Java 2.x und dem SDK for Python auflisten.

SDK for Java 2.x

```
public static void listMultiPartUploadsParts( S3Client s3, String bucketName, String
 objKey, String uploadID) {

        try {
            ListPartsRequest listPartsRequest = ListPartsRequest.builder()
                .bucket(bucketName)
                .uploadId(uploadID)
                .key(objKey)
                .build();

            ListPartsResponse response = s3.listParts(listPartsRequest);
            ListPart parts = response.parts();
            for (Part part: parts) {
                System.out.println("Upload in progress: Part number = \"" +
 part.partNumber() + "\", etag = " + part.eTag());
            }

        }

        catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }


    }
```

SDK for Python

```
import logging
import boto3
from botocore.exceptions import ClientError

def list_parts(s3_client, bucket_name, key_name, upload_id):
    '''
```

```
    Lists the parts that have been uploaded for a specific multipart upload to a
 directory bucket.

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket that multipart uploads parts have been uploaded to
    :param key_name: Name of the object that has parts uploaded
    :param upload_id: Multipart upload ID that the parts are associated with
    :return: List of parts associated with the specified multipart upload, None if
 there are no parts
    '''
    parts_list = []
    next_part_marker = ''
    continuation_flag = True
    try:
        while continuation_flag:
            if next_part_marker == '':
                response = s3_client.list_parts(
                    Bucket = bucket_name,
                    Key = key_name,
                    UploadId = upload_id
                )
            else:
                response = s3_client.list_parts(
                    Bucket = bucket_name,
                    Key = key_name,
                    UploadId = upload_id,
                    NextPartMarker = next_part_marker
                )
            if 'Parts' in response:
                for part in response['Parts']:
                    parts_list.append(part)
                if response['IsTruncated']:
                    next_part_marker = response['NextPartNumberMarker']
                else:
                    continuation_flag = False
            else:
                continuation_flag = False
        return parts_list
    except ClientError as e:
        logging.error(e)
        return None

if __name__ == '__main__':
    region = 'us-west-2'
```

```
    bucket_name = 'BUCKET_NAME'
    key_name = 'KEY_NAME'
    upload_id = 'UPLOAD_ID'
    s3_client = boto3.client('s3', region_name = region)
    parts_list = list_parts(s3_client, bucket_name, key_name, upload_id)
    if parts_list is not None:
        print (f'{key_name} has {len(parts_list)} parts uploaded to {bucket_name}')
    else:
        print (f'There are no multipart uploads with that upload ID for
{bucket_name} bucket')
```

Hochladen eines Objekts zu einem Verzeichnis-Bucket

Die Kopieroperation erzeugt eine Kopie eines Objekts, das bereits in Amazon S3 gespeichert ist. Sie können Objekte zwischen Verzeichnis-Buckets und Allzweck-Buckets kopieren. Sie können Objekte auch innerhalb eines Buckets und zwischen Buckets desselben Typs kopieren, z. B. von Verzeichnis-Bucket zu Verzeichnis-Bucket.

Sie können eine Kopie eines Objekts bis zu 5 GB in einer einzigen atomaren Operation erstellen. Um jedoch ein Objekt zu kopieren, das größer als 5 GB ist, müssen Sie die API-Operationen für mehrteilige Uploads verwenden. Weitere Informationen finden Sie unter Verwenden von mehrteiligen Uploads mit Verzeichnis-Buckets.

Berechtigungen

Um Objekte zu kopieren, müssen Sie über die folgenden Berechtigungen verfügen:

- Um Objekte von einem Verzeichnis-Bucket zu einem anderen Verzeichnis-Bucket zu kopieren, benötigen Sie die `s3express:CreateSession`-Berechtigung.
- Um Objekte aus Verzeichnis-Buckets zu Allzweck-Buckets zu kopieren, benötigen Sie die `s3express:CreateSession`-Berechtigung und `s3:PutObject`-Berechtigung zum Schreiben der Objektkopie in den Ziel-Bucket.
- Um Objekte aus Allzweck-Buckets in Verzeichnis-Buckets zu kopieren, benötigen Sie die `-s3express:CreateSession`Berechtigung und die `-s3:GetObject`Berechtigung zum Lesen des Quellobjekts, das kopiert wird.

  Weitere Informationen finden Sie unter CopyObject in der API-Referenz zu Amazon Simple Storage Service.

Verschlüsselung

Amazon S3 verschlüsselt automatisch alle neuen Objekte, die in einen S3-Bucket hochgeladen werden. Die Standardverschlüsselungskonfiguration eines S3-Buckets ist immer aktiviert und mindestens auf serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) eingestellt.

Für Verzeichnis-Buckets wird nur SSE-S3 unterstützt. Für Allzweck-Buckets können Sie SSE-S3 (Standard), serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) oder serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden.

Wenn Sie eine Kopieranforderung stellen, die die Parameter SSE-C, SSE-KMS oder DSSE-KMS in einem Verzeichnis-Bucket als Quelle oder Ziel festlegt, gibt die Antwort einen Fehler zurück.

Tags

Verzeichnis-Buckets unterstützen keine Tags. Wenn Sie ein Objekt mit Tags aus einem Allzweck-Bucket in einen Verzeichnis-Bucket kopieren, erhalten Sie eine HTTP-`501 (Not Implemented)`Antwort. Weitere Informationen finden Sie unter CopyObject in der API-Referenz zu Amazon Simple Storage Service.

ETags

Entitäts-Tags (ETags) für S3 Express One Zone sind zufällige alphanumerische Zeichenfolgen und keine MD5-Prüfsummen. Verwenden Sie zusätzliche Prüfsummen, um die Objektintegrität zu gewährleisten.

Zusätzliche Prüfsummen

S3 Express One Zone bietet Ihnen die Möglichkeit, den Prüfsummenalgorithmus auszuwählen, der zur Validierung Ihrer Daten beim Hoch- oder Herunterladen verwendet wird. Sie können einen der folgenden Secure Hash Algorithms (SHA)- oder Cyclic Redundancy Check (CRC)-Algorithmen zur Überprüfung der Datenintegrität auswählen: CRC32, CRC32C, SHA-1 und SHA-256. MD5-based Prüfsummen werden von der Speicherklasse S3 Express One Zone nicht unterstützt.

Weitere Informationen finden Sie unter Bewährte Methoden für zusätzliche S3-Prüfsummen.

Unterstützte Features

Weitere Informationen darüber, welche Amazon S3-Funktionen für S3 Express One Zone unterstützt werden, finden Sie unter Wodurch zeichnet sich S3 Express One Zone aus?.

Verwenden der S3-Konsole (Kopieren in einen Verzeichnis-Bucket)

So kopieren Sie ein Objekt aus einem Allzweck-Bucket oder einem Verzeichnis-Bucket in einen Verzeichnis-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie den Bucket aus, aus dem Sie Objekte kopieren möchten:
   - Um aus einem Allzweck-Bucket zu kopieren, wählen Sie die Registerkarte Allzweck-Buckets.
   - Um aus einem Verzeichnis-Bucket zu kopieren, wählen Sie die Registerkarte Verzeichnis-Buckets.
4. Wählen Sie den Allzweck-Bucket oder Verzeichnis-Bucket aus, der die Objekte enthält, die Sie kopieren möchten.
5. Wählen Sie dieObjekte-Tag. Aktivieren Sie auf der Seite Objekte das Kontrollkästchen links neben den Namen der Objekte, die Sie kopieren möchten.
6. Wählen Sie im Menü Actions (Aktionen) die Option Copy (Kopieren) aus.

   Die Seite Kopieren wird angezeigt.
7. Wählen Sie unter Ziel die Option Verzeichnis-Bucket für Ihren Zieltyp aus. Um den Zielpfad anzugeben, wählen Sie S3 durchsuchen, navigieren Sie zum Ziel und wählen Sie dann die Optionsschaltfläche links neben dem Ziel. Wählen Sie unten rechts Choose destination (Ziel auswählen) aus.

   Geben Sie alternativ den Zielpfad ein.
8. Wählen Sie unter Prüfsummen aus, ob Sie die Objekte mit ihren vorhandenen Prüfsummenfunktionen kopieren oder die vorhandenen Prüfsummenfunktionen durch eine neue ersetzen möchten. Beim Hochladen der Objekte hatten Sie die Möglichkeit, den Prüfsummenalgorithmus anzugeben, der zur Überprüfung der Datenintegrität verwendet wurde. Beim Kopieren des Objekts haben Sie die Möglichkeit, eine neue Funktion auszuwählen. Wenn Sie ursprünglich keine zusätzliche Prüfsumme angeben, können Sie den Abschnitt e Prüfsummen verwenden, um eine hinzuzufügen.

> **Note**
>
> Selbst wenn Sie dieselbe Prüfsummenfunktion verwenden, kann sich Ihr Prüfsummenwert ändern, wenn das Objekt größer als 16 MB ist. Der Prüfsummenwert kann sich aufgrund der Methode ändern, wie Prüfsummen für mehrteilige Uploads berechnet werden. Weitere Informationen dazu, wie sich die Prüfsumme beim Kopieren des Objekts ändern kann, finden Sie unter Verwenden von Prüfsummen auf Teilebene für mehrteilige Uploads.

Um die Prüfsummenfunktion zu ändern, wählen Sie Replace with a new checksum function (Durch eine neue Prüfsummenfunktion ersetzen) aus. Wählen Sie die neue Prüfsummenfunktion aus der Dropdown-Liste aus. Wenn das Objekt kopiert wird, wird die neue Prüfsumme mit dem angegebenen Algorithmus berechnet und gespeichert.

9. Wählen Sie unten rechts Copy (Kopieren) aus. Amazon S3 kopiert Ihre Objekte in den Zielordner.

Verwenden der S3-Konsole (Kopieren zu einem Allzweck-Bucket)

So kopieren Sie ein Objekt aus einem Verzeichnis-Bucket in einen Allzweck-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie die Registerkarte Verzeichnis-Buckets aus.
4. Wählen Sie den Verzeichnis-Bucket aus, der die Objekte enthält, die Sie kopieren möchten.
5. Wählen Sie dieObjekte-Tag. Aktivieren Sie auf der Seite Objekte das Kontrollkästchen links neben den Namen der Objekte, die Sie kopieren möchten.
6. Wählen Sie im Menü Actions (Aktionen) die Option Copy (Kopieren) aus.
7. Wählen Sie unter Ziel die Option Allzweck-Bucket für Ihren Zieltyp aus. Um den Zielpfad anzugeben, wählen Sie S3 durchsuchen, navigieren Sie zum Ziel und wählen Sie die Optionsschaltfläche links neben dem Ziel. Wählen Sie unten rechts Choose destination (Ziel auswählen) aus.

Geben Sie alternativ den Zielpfad ein.

8. Wählen Sie unter Prüfsummen aus, ob Sie die Objekte mit ihren vorhandenen Prüfsummenfunktionen kopieren oder die vorhandenen Prüfsummenfunktionen durch eine neue ersetzen möchten. Beim Hochladen der Objekte hatten Sie die Möglichkeit, den Prüfsummenalgorithmus anzugeben, der zur Überprüfung der Datenintegrität verwendet wurde. Beim Kopieren des Objekts haben Sie die Möglichkeit, eine neue Funktion auszuwählen. Wenn Sie ursprünglich keine zusätzliche Prüfsumme angegeben haben, können Sie im Abschnitt Prüfsummen eine hinzufügen.

   > **Note**
   >
   > Selbst wenn Sie dieselbe Prüfsummenfunktion verwenden, kann sich Ihr Prüfsummenwert ändern, wenn das Objekt größer als 16 MB ist. Der Prüfsummenwert kann sich aufgrund der Methode ändern, wie Prüfsummen für mehrteilige Uploads berechnet werden. Weitere Informationen dazu, wie sich die Prüfsumme beim Kopieren des Objekts ändern kann, finden Sie unter Verwenden von Prüfsummen auf Teilebene für mehrteilige Uploads.

   Um die Prüfsummenfunktion zu ändern, wählen Sie Replace with a new checksum function (Durch eine neue Prüfsummenfunktion ersetzen) aus. Wählen Sie die neue Prüfsummenfunktion aus der Dropdown-Liste aus. Wenn das Objekt kopiert wird, wird die neue Prüfsumme mit dem angegebenen Algorithmus berechnet und gespeichert.

9. Wählen Sie unten rechts Copy (Kopieren) aus. Amazon S3 kopiert Ihre Objekte in den Zielordner.

Verwenden der AWS SDKs

SDK for Java 2.x

Example

```
public static void copyBucketObject (S3Client s3, String sourceBucket, String
objectKey, String targetBucket) {
     CopyObjectRequest copyReq = CopyObjectRequest.builder()
         .sourceBucket(sourceBucket)
         .sourceKey(objectKey)
```

```
            .destinationBucket(targetBucket)
            .destinationKey(objectKey)
            .build();
        String temp = "";

        try {
            CopyObjectResponse copyRes = s3.copyObject(copyReq);
            System.out.println("Successfully copied " + objectKey +" from bucket " +
sourceBucket +" into bucket "+targetBucket);
        }

        catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
}
```

Löschen eines Objekts in einem Verzeichnis-Bucket

Sie können Objekte aus einem Amazon S3-Verzeichnis-Bucket löschen, indem Sie die Amazon S3-Konsole, AWS Command Line Interface (AWS CLI) oder AWS SDKs verwenden. SDKs Weitere Informationen finden Sie unter Verzeichnis-Buckets und Was ist S3 Express One Zone?.

Warning

- Das Löschen eines Objekts kann nicht rückgängig gemacht werden.
- Diese Aktion löscht alle angegebenen Objekte. Warten Sie beim Löschen von Ordnern, bis die Löschaktion abgeschlossen ist, bevor Sie dem Ordner neue Objekte hinzufügen. Andernfalls könnten auch neue Objekte gelöscht werden.

Note

Wenn Sie mehrere Objekte programmgesteuert aus einem Verzeichnis-Bucket löschen, beachten Sie Folgendes:

- Objektschlüssel in `DeleteObjects`-Anforderungen müssen mindestens ein Zeichen enthalten, das kein Leerzeichen ist. Zeichenfolgen aller Leerzeichen werden nicht unterstützt.

- Objektschlüssel in -`DeleteObjects`Anforderungen dürfen keine Unicode-Steuerzeichen enthalten, mit Ausnahme von Zeilenumbruch (`\n`), Tabulator (`\t`) und Zeilenumschaltung (`\r`).

Verwenden der S3-Konsole

So löschen Sie Objekte

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie die Registerkarte Verzeichnis-Buckets.
4. Wählen Sie den Verzeichnis-Bucket aus, der die Objekte enthält, die Sie löschen möchten.
5. Wählen Sie dieObjekte-Tag. Aktivieren Sie in der Liste Objekte das Kontrollkästchen links neben dem Objekt oder den Objekten, die Sie löschen möchten.
6. Wählen Sie Löschen aus.
7. Geben Sie auf der Seite Objekte löschen **`permanently delete`** in das Textfeld ein.
8. Wählen Sie Delete objects (Objekte löschen).

Verwenden der AWS-SDKs

SDK for Java 2.x

Example

Im folgenden Beispiel werden Objekte in einem Verzeichnis-Bucket mithilfe der gelöschtAWS SDK for Java 2.x.

```
static void deleteObject(S3Client s3Client, String bucketName, String objectKey) {



       try {

           DeleteObjectRequest del = DeleteObjectRequest.builder()
                   .bucket(bucketName)
```

```
                    .key(objectKey)
                    .build();

            s3Client.deleteObject(del);

            System.out.println("Object " + objectKey + " has been deleted");


        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

    }
```

SDK for Python

Example

Im folgenden Beispiel werden Objekte in einem Verzeichnis-Bucket mithilfe der gelöschtAWS SDK for Python (Boto3).

```
import logging
import boto3
from botocore.exceptions import ClientError

def delete_objects(s3_client, bucket_name, objects):
    '''
    Delete a list of objects in a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket that contains objects to be deleted; for example,
 'doc-example-bucket--usw2-az1--x-s3'
    :param objects: List of dictionaries that specify the key names to delete
    :return: Response output, else False
    '''

    try:
        response = s3_client.delete_objects(
            Bucket = bucket_name,
            Delete = {
                'Objects': objects
```

```
            }
        )
        return response
    except ClientError as e:
        logging.error(e)
        return False


if __name__ == '__main__':
    region = 'us-west-2'
    bucket_name = 'BUCKET_NAME'
    objects = [
        {
            'Key': '0.txt'
        },
        {
            'Key': '1.txt'
        },
        {
            'Key': '2.txt'
        },
        {
            'Key': '3.txt'
        },
        {
            'Key': '4.txt'
        }
    ]

    s3_client = boto3.client('s3', region_name = region)
    results = delete_objects(s3_client, bucket_name, objects)
    if results is not None:
        if 'Deleted' in results:
            print (f'Deleted {len(results["Deleted"])} objects from {bucket_name}')
        if 'Errors' in results:
            print (f'Failed to delete {len(results["Errors"])} objects from
 {bucket_name}')
```

Herunterladen eines Objekts in einem Verzeichnis-Bucket

Die folgenden Codebeispiele zeigen, wie Sie mithilfe der -`GetObject`API-Operation Daten aus einem Objekt in einem Amazon S3-Verzeichnis-Bucket lesen (herunterladen).

Verwenden der AWS-SDKs

SDK for Java 2.x

Example

Das folgende Codebeispiel zeigt, wie Daten aus einem Objekt in einem Verzeichnis-Bucket mithilfe der gelesen werdenAWS SDK for Java 2.x.

```
public static void getObject(S3Client s3Client, String bucketName, String objectKey)
 {
     try {
         GetObjectRequest objectRequest = GetObjectRequest
            .builder()
            .key(objectKey)
            .bucket(bucketName)
            .build();

         ResponseBytes GetObjectResponse objectBytes =
 s3Client.getObjectAsBytes(objectRequest);
         byte[] data = objectBytes.asByteArray();

         //Print object contents to console
         String s = new String(data, StandardCharsets.UTF_8);
         System.out.println(s);
    }

    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

SDK for Python

Example

Das folgende Codebeispiel zeigt, wie Daten aus einem Objekt in einem Verzeichnis-Bucket mithilfe der gelesen werdenAWS SDK for Python (Boto3).

```
import boto3
from botocore.exceptions import ClientError
```

```
from botocore.response import StreamingBody

def get_object(s3_client: boto3.client, bucket_name: str, key_name: str) ->
 StreamingBody:
    """
    Gets the object.
    :param s3_client:
    :param bucket_name: The bucket that contains the object.
    :param key_name: The key of the object to be downloaded.
    :return: The object data in bytes.
    """
    try:
        response = s3_client.get_object(Bucket=bucket_name, Key=key_name)
        body = response['Body'].read()
        print(f"Got object '{key_name}' from bucket '{bucket_name}'.")
    except ClientError:
        print(f"Couldn't get object '{key_name}' from bucket '{bucket_name}'.")
        raise
    else:
        return body

def main():
    s3_client = boto3.client('s3')
    resp = get_object(s3_client, 'doc-example-bucket--use1-az4--x-s3', 'sample.txt')
    print(resp)

if __name__ == "__main__":
     main()
```

Verwenden von **`HeadObject`** mit Verzeichnis-Buckets

Das folgende AWS SDK-Beispiel zeigt, wie Sie die `HeadObject`-API-Operation verwenden, um Metadaten aus einem Objekt in einem Amazon S3-Verzeichnis-Bucket abzurufen, ohne das Objekt selbst zurückzugeben.

Verwenden der AWS-SDKs

SDK for Java 2.x

Example

```
public static void headObject(S3Client s3Client, String bucketName, String
 objectKey) {
    try {
        HeadObjectRequest headObjectRequest = HeadObjectRequest
                .builder()
                .bucket(bucketName)
                .key(objectKey)
                .build();
        HeadObjectResponse response = s3Client.headObject(headObjectRequest);
        System.out.format("Amazon S3 object: \"%s\" found in bucket: \"%s\" with
 ETag: \"%s\"", objectKey, bucketName, response.eTag());
    }
    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
```

Sicherheit für S3 Express One Zone

Die Sicherheit in der Cloud hat für AWS höchste Priorität. Als AWS-Kunde profitieren Sie von Rechenzentren und Netzwerkarchitekturen, die eingerichtet wurden, um die Anforderungen der anspruchsvollsten Organisationen in puncto Sicherheit zu erfüllen. Sicherheit gilt zwischen AWS und Ihnen eine geteilte Verantwortung. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud selbst – AWS ist dafür verantwortlich, die Infrastruktur zu schützen, mit der AWS-Services in der AWS Cloud ausgeführt werden. AWS stellt Ihnen außerdem Services bereit, die Sie sicher nutzen können. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der AWS Compliance Programs-Compliance-Programme regelmäßig.

  Weitere Informationen zu den für Amazon S3 Express One Zone geltenden Compliance-Programmen finden Sie unter AWS-Services in Scope by Compliance Program.
- Sicherheit in der Cloud – Ihr Verantwortungsumfang wird durch den AWS-Service bestimmt, den Sie verwenden. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, die Anforderungen Ihres Unternehmens und die geltenden Gesetze und Vorschriften.

Diese Dokumentation beschreibt, wie Sie das Modell der gemeinsamen Verantwortlichkeit bei der Verwendung von S3 Express Zone anwenden können. Die folgenden Themen veranschaulichen, wie Sie S3 Express One Zone zur Erfüllung Ihrer Sicherheits- und Compliance-Anforderungen

konfigurieren können. Sie erfahren außerdem, wie Sie andere AWS-Services verwenden können, die Ihnen beim Überwachen und Sichern Ihrer Ressourcen helfen, wenn Sie mit S3 Express One Zone arbeiten.

Themen

Datenschutz und Verschlüsselung

Weitere Informationen darüber, wie S3 Express One Zone Ihre Daten verschlüsselt und schützt, finden Sie in den folgenden Themen.

Themen

Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)

Standardmäßig werden alle in Verzeichnis-Bucket gespeicherten Objekte automatisch mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) verschlüsselt. Unverschlüsselte Uploads zu Verzeichnis-Buckets sind nicht zulässig. Weitere Informationen finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) und Datenschutz durch Verschlüsselung.

Verzeichnis-Buckets unterstützen keine serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), die serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS) oder die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C).

Verschlüsselung während der Übertragung

Auf S3 Express One Zone kann nur über HTTPS (TLS) zugegriffen werden.

S3 Express One Zone verwendet regionale und zonale API-Endpunkte. Je nachdem, welche Amazon-S3-API-Operation Sie verwenden, ist entweder ein regionaler oder ein zonaler Endpunkt erforderlich. Sie können über einen Gateway Virtual Private Cloud (VPC)-Endpunkt auf zonale und regionale Endpunkte zugreifen. Für die Nutzung von Gateway-Endpunkten fallen keine zusätzlichen Gebühren an. Weitere Informationen zu regionalen und zonalen API-Endpunkten finden Sie unter Networking für S3 Express One Zone.

Zusätzliche Prüfsummen

S3 Express One Zone bietet Ihnen die Möglichkeit, den Prüfsummenalgorithmus auszuwählen, der zur Validierung Ihrer Daten beim Hoch- oder Herunterladen verwendet wird. Sie können einen der folgenden Secure Hash Algorithms (SHA)- oder Cyclic Redundancy Check (CRC)-Algorithmen zur Überprüfung der Datenintegrität auswählen: CRC32, CRC32C, SHA-1 und SHA-256. MD5-based Prüfsummen werden von der Speicherklasse S3 Express One Zone nicht unterstützt.

Weitere Informationen finden Sie unter Bewährte Methoden für zusätzliche S3-Prüfsummen.

Löschen von Daten

Sie können ein oder mehrere Objekte direkt aus S3 Express One Zone mit der Amazon-S3-Konsole, AWS-SDKs, AWS Command Line Interface (AWS CLI) oder REST-API löschen. Für alle Objekte in Ihrem S3-Bucket entstehen Speicherkosten, deshalb sollten Sie Objekte löschen, die Sie nicht mehr benötigen.

Beim Löschen eines Objekts, das in einem Verzeichnis-Bucket gespeichert ist, werden auch alle übergeordneten Verzeichnisse rekursiv gelöscht, sofern diese übergeordneten Verzeichnisse keine anderen Objekte als das Objekt enthalten, das gelöscht wird.

> **Note**
>
> Das Löschen mit Multi-Faktor-Authentifizierung (MFA) und S3 Versioning werden für S3 Express One Zone nicht unterstützt.

AWS Identity and Access Management (IAM) für S3 Express One Zone

AWS Identity and Access Management (IAM) ist ein AWS-Service , mit dem Administratoren den Zugriff auf - AWS Ressourcen sicher steuern können. IAM-Administratoren steuern, wer authentifiziert (angemeldet) und autorisiert (im Besitz von Berechtigungen) ist, Amazon-S3-Ressourcen in S3 Express One Zone zu nutzen. Sie können IAM ohne zusätzliche Kosten nutzen.

Standardmäßig haben Benutzer keine Berechtigungen für Verzeichnis-Buckets und S3-Express-One-Zone-Vorgänge. Um Zugriffsberechtigungen für Verzeichnis-Buckets zu gewähren, können Sie IAM verwenden, um Benutzer, Gruppen oder Rollen zu erstellen und diesen Identitäten Berechtigungen zuzuweisen. Weitere Informationen finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Um Zugriff zu gewähren, können Sie Ihren Benutzern, Gruppen oder Rollen auf die folgenden Weisen Berechtigungen hinzufügen:

- Benutzer und Gruppen in AWS IAM Identity Center – Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter Erstellen eines Berechtigungssatzes im AWS IAM Identity Center -Benutzerhandbuch.
- In IAM über einen Identitätsanbieter verwaltete Benutzer – Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter Erstellen einer Rolle für einen externen Identitätsanbieter (Verbund) im IAM-Benutzerhandbuch.
- IAM-Rollen und -Benutzer – Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Befolgen Sie die Anweisungen in Erstellen einer Rolle, um Berechtigungen an einen IAM-Benutzer zu delegieren im IAM-Benutzerhandbuch.

Standardmäßig sind Verzeichnis-Buckets privat und sind nur für Benutzer zugänglich, denen explizit Zugriff gewährt wurde. Die Zugriffskontrollgrenze für Verzeichnis-Buckets wird nur auf Bucket-Ebene festgelegt. Im Gegensatz dazu kann die Zugriffskontrollgrenze für Allzweck-Buckets auf Bucket-, Präfix- oder Objekt-Tag-Ebene festgelegt werden. Dieser Unterschied bedeutet, dass Verzeichnis-Buckets die einzige Ressource sind, die Sie in Bucket-Richtlinien oder IAM-Identitätsrichtlinien für den Zugriff auf S3 Express One Zone aufnehmen können.

Mit S3 Express One Zone authentifizieren und autorisieren Sie Anforderungen zusätzlich zur IAM-Autorisierung über einen neuen sitzungsbasierten Mechanismus, der von der `CreateSession`-API-Operation abgewickelt wird. Sie können mit `CreateSession` temporäre Anmeldeinformationen anfordern, die den Zugriff auf Ihren Bucket mit geringer Latenz ermöglichen. Diese temporären Anmeldeinformationen sind auf einen bestimmten Verzeichnis-Bucket beschränkt.

Um mit zu arbeiten`CreateSession`, empfehlen wir, die neueste Version der - AWS SDKs oder die AWS Command Line Interface () zu verwenden AWS CLI. Die AWS SDKs-SDKs und die AWS CLI übernehmen die Einrichtung, Aktualisierung und Beendigung von Sitzungen in Ihrem Namen.

Sie verwenden Sitzungstoken nur für zonale Operationen (Objektebene) (außer `CopyObject` und `HeadBucket`), um die mit der Autorisierung verbundene Latenz auf eine Reihe von Anforderungen in einer Sitzung zu verteilen. Für API-Operationen auf regionalen Endpunkten (Operationen auf Bucket-Ebene) verwenden Sie die IAM-Autorisierung, bei der keine Sitzung verwaltet wird. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone und `CreateSession`-Autorisierung.

Weitere Informationen zu IAM für S3 Express One Zone finden Sie in den folgenden Themen.

Themen

- Auftraggeber
- Ressourcen
- Aktionen für S3 Express One Zone
- Bedingungsschlüssel für S3 Express One Zone
- Wie API-Vorgänge autorisiert und authentifiziert werden

Auftraggeber

Wenn Sie eine ressourcenbasierte Richtlinie erstellen, um Zugriff auf Ihre Buckets zu gewähren, müssen Sie das `Principal`-Element verwenden, um die Person oder Anwendung anzugeben, die eine Anforderung für eine Aktion oder einen Vorgang auf dieser Ressource stellen kann. Für Verzeichnis-Bucket-Richtlinien können Sie die folgenden Prinzipale verwenden:

- Ein - AWS Konto
- Ein IAM-Benutzer
- Eine IAM-Rolle
- Ein Verbundbenutzer

Weitere Informationen finden Sie unter Principal im IAM-Benutzerhandbuch.

Ressourcen

Amazon-Ressourcennamen (ARNs) für Verzeichnis-Buckets enthalten den `s3express` Namespace, die AWS-Region, die AWS Konto-ID und den Namen des Verzeichnis-Buckets, der die Availability Zone-ID enthält. Wenn Sie auf Ihren Verzeichnis-Bucket zugreifen und Aktionen für diesen ausführen möchten, müssen Sie das folgende ARN-Format verwenden:

```
arn:aws:s3express:region:account-id:bucket/base-bucket-name--azid--x-s3
```

Weitere Informationen finden Sie unter Amazon Resource Names (ARNs) im IAM-Benutzerhandbuch. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Resource im IAM-Benutzerhandbuch.

Aktionen für S3 Express One Zone

In einer auf IAM-Identitäten oder auf Ressourcen basierenden Richtlinie legen Sie fest, welche S3--Aktionen zugelassen sind oder oder abgelehnt werden. S3-Express-One-Zone-Aktionen entsprechen bestimmten API-Vorgängen. S3 Express One Zone hat einen eindeutigen IAM-Namespace, der sich vom Standard-Namespace für Amazon S3 unterscheidet. Dieser Namespace ist `s3express`.

Wenn Sie die `s3express:CreateSession`-Berechtigung zulassen, ermöglicht dies dem `CreateSession`-API-Vorgang, Sitzungstoken abzurufen, wenn auf API-Operationen mit zonalen Endpunkten (oder auf Objektebene) zugegriffen wird. Diese Sitzungstoken geben Anmeldeinformationen zurück, die verwendet werden, um Zugriff auf alle anderen API-Operationen für zonale Endpunkte zu gewähren. Daher müssen Sie nicht mithilfe von IAM-Richtlinien Zugriffsberechtigungen für zonale API-Operationen gewähren. Stattdessen ermöglicht das Sitzungstoken den Zugriff.

Weitere Informationen über API-Operationen für zonale und regionale Endpunkte finden Sie unter Networking für S3 Express One Zone. Weitere Informationen zum `CreateSession`-API-Vorgang finden Sie unter CreateSession in der Amazon-Simple-Storage-Service-API-Referenz.

Sie können die folgenden Aktionen im Element `Action` einer IAM-Richtlinienanweisung angeben. Verwenden Sie Richtlinien, um Berechtigungen zum Ausführen einer Operation in AWS zu erteilen. Wenn Sie eine Aktion in einer Richtlinie verwenden, gestatten oder verweigern Sie in der Regel den Zugriff auf den API-Vorgang mit demselben Namen. Dabei kann es mitunter vorkommen, dass eine einzige Aktion den Zugriff auf mehr als eine API-Operation steuert. Der Zugriff auf Aktionen auf Bucket-Ebene kann nur über identitätsbasierte IAM-Richtlinien (Benutzer oder Rolle) und nicht über Bucket-Richtlinien gewährt werden.

Aktionen und Bedingungsschlüssel für S3 Express One Zone

Aktion	API	Beschreibung	Zugriffsebene	Bedingungsschlüssel
`s3express:CreateBucket`	`CreateBucket`	Gewährt die Berechtigung zum Erstellen eines neuen Buckets.	Schreibe	`s3express:authType` `s3express:LocationName` `s3express:ResourceAccount` `s3express:signatureversion` `s3express:TlsVersion` `s3express:x-amz-content-sha256`
`s3express:CreateSession`	`CreateSession`	Gewährt die Berechtigung zum Erstellen eines Sitzungstokens, mit dem Zugriff auf alle zonalen API-Operationen (auf Objektebene) gewährt wird, z. B. `PutObject` , `GetObject` usw.	Schreibe	`s3express:authType` `s3express:SessionMode` `s3express:ResourceAccount`

Aktion	API	Beschreibung	Zugriffsebene	Bedingungsschlüssel
				s3express:signatureversion s3express:signatureAge s3express:TlsVersion s3express:x-amz-content-sha256
s3express:DeleteBucket	DeleteBucket	Gewährt die Berechtigung zum Löschen des im URI genannten Buckets.	Schreibe	s3express:authType s3express:ResourceAccount s3express:signatureversion s3express:TlsVersion s3express:x-amz-content-sha256

Aktion	API	Beschreibung	Zugriffsebene	Bedingungsschlüssel
`s3express:DeleteBucketPolicy`	`DeleteBucketPolicy`	Gewährt die Berechtigung zum Löschen der Richtlinie für einen angegebenen Bucket.	Berechtigungsverwaltung	`s3express:authType` `s3express:ResourceAccount` `s3express:signatureversion` `s3express:TlsVersion` `s3express:x-amz-content-sha256`

Aktion	API	Beschreibung	Zugriffsebene	Bedingungsschlüssel
`s3express:GetBucketPolicy`	`GetBucketPolicy`	Gewährt die Berechtigung zum Zurückgeben der Richtlinie des angegebenen Buckets.	Lesen	`s3express:authType` `s3express:ResourceAccount` `s3express:signatureversion` `s3express:TlsVersion` `s3express:x-amz-content-sha256`

Aktion	API	Beschreibung	Zugriffsebene	Bedingungsschlüssel
`s3express:ListAllMyDirectoryBuckets`	`ListDirectoryBuckets`	Gewährt die Berechtigung zum Auflisten aller Verzeichnis-Buckets, die dem authentifizierten Sender der Anforderung gehören.	Auflisten	`s3express:authType` `s3express:ResourceAccount` `s3express:signatureversion` `s3express:TlsVersion` `s3express:x-amz-content-sha256`

Aktion	API	Beschreibung	Zugriffsebene	Bedingungsschlüssel
`s3express:PutBucketPolicy`	`PutBucketPolicy`	Gewährt die Berechtigung zum Hinzufügen oder Ersetzen einer Bucket-Richtlinie für einen Bucket.	Berechtigungsverwaltung	`s3express:authType` `s3express:ResourceAccount` `s3express:signatureversion` `s3express:TlsVersion` `s3express:x-amz-content-sha256`

Bedingungsschlüssel für S3 Express One Zone

S3 Express One Zone definiert die folgenden Bedingungsschlüssel, die im `Condition`-Element einer IAM-Richtlinie verwendet werden können. Diese Schlüssel können Sie verwenden, um die Bedingungen zu verfeinern, unter denen die Richtlinienanweisung angewendet wird.

Bedingungsschlüssel	Beschreibung	Typ
`s3express:authType`	Filtert den Zugriff nach Authentifizierungsmethode Um eingehende Anfragen auf die Verwendung einer bestimmten Authentifizierungsmethode zu beschränken, können Sie diesen optionalen Bedingungsschlüssel verwenden. Sie können diesen Bedingung	String

Bedingungsschlüssel	Beschreibung	Typ
	sschlüssel zum Beispiel verwenden, um nur den HTTP-`Authorization` -Header für die Authentifizierung von Anfragen zuzulassen. Zulässige Werte: `REST-HEADER` , `REST-QUERY-STRING`	
`s3express:LocationName`	Filtert den Zugriff auf den `CreateBucket` -.API-Vorgang nach einer bestimmten Availability Zone-ID (AZ-ID), z. B. `usw2-az1`. Beispielwert: `usw2-az1`	String
`s3express:ResourceAccount`	Filtert den Zugriff nach der AWS-Konto ID des Ressourcenbesitzers. Um den Benutzer-, Rollen- oder Anwendungszugriff auf die Verzeichnis-Buckets einzuschränken, die einer bestimmten AWS-Konto ID gehören, können Sie entweder den `s3express:ResourceAccount` Bedingungsschlüssel `aws:ResourceAccount` oder verwenden. Sie können diesen Bedingungsschlüssel entweder in AWS Identity and Access Management (IAM)-Identitätsrichtlinien oder Virtual Private Cloud (VPC)-Endpunktrichtlinien verwenden. Sie können diesen Bedingungsschlüssel beispielsweise verwenden, um Clients innerhalb Ihrer VPC daran zu hindern, auf Buckets zuzugreifen, die Sie nicht besitzen. Beispielwert: `111122223333`	String

Bedingungsschlüssel	Beschreibung	Typ
`s3express:SessionMode`	Filtert den Zugriff nach der vom `CreateSession` -API-Vorgang angeforderten Berechtigung. Standardmäßig ist die Sitzung auf `ReadWrite` eingestellt. Sie können diesen Bedingungsschlüssel verwenden, um den Zugriff auf `ReadOnly` zu beschränken oder den `ReadWrite` -Zugriff explizit zu verweigern. Weitere Informationen finden Sie unter Beispiel für Verzeichnis-Bucket-Richtlinien für S3 Express One Zone und unter CreateSession in der API-Referenz für Amazon Simple Storage Service. Zulässige Werte: `ReadWrite` , `ReadOnly`	String
`s3express:signatureAge`	Filtert den Zugriff nach dem Alter der Anforderungssignatur in Millisekunden. Diese Bedingung gilt nur für vorsignierte URLs. In AWS Signature Version 4 ist der Signaturschlüssel bis zu sieben Tage gültig. Daher sind die Signaturen auch bis zu sieben Tage lang gültig. Weitere Informationen finden Sie unter Einführung in das Signieren von Anfragen in der Amazon Simple Storage Service API-Referenz. Sie können diese Bedingung verwenden, um das Alter der Unterschrift weiter einzuschränken. Beispielwert: `600000`	Numerischer Wert

Bedingungsschlüssel	Beschreibung	Typ
`s3express:signaturversion`	Identifiziert die Version von AWS Signature, die Sie für authentifizierte Anforderungen unterstützen möchten. Für authentifizierte Anfragen unterstützt S3 Express One Zone Signature Version 4. Gültiger Wert: `"AWS4-HMAC-SHA256"` (identifiziert Signature Version 4)	String
`s3express:TlsVersion`	Filtert den Zugriff nach der TLS-Version, die vom Client verwendet wird Sie können den `-s3:TlsVersion` Bedingungsschlüssel verwenden, um IAM-, Virtual Private Cloud Endpoint (VPCE)- oder Bucket-Richtlinien zu schreiben, die den Benutzer- oder Anwendungszugriff auf Verzeichnis-Buckets basierend auf der TLS-Version einschränken, die vom Client verwendet wird. Sie können diesen Bedingungsschlüssel auch verwenden, um Richtlinien zu schreiben, die eine minimale TLS-Version erfordern. Beispielwert: `1.3`	Numerischer Wert

Bedingungsschlüssel	Beschreibung	Typ
`s3express:x-amz-content-sha256`	Filtert den Zugriff auf nicht signierte Inhalte in Ihrem Bucket. Sie können diesen Bedingungsschlüssel verwenden, um nicht signierte Inhalte in Ihrem Bucket zu verbieten. Wenn Sie Signature Version 4 verwenden, fügen Sie bei Anfragen, die den `Authorization` -Header verwenden, den `x-amz-content-sha256` -Header in die Signaturberechnung ein und setzen dann dessen Wert auf die Hash-Nutzlast. Sie können diesen Bedingungsschlüssel in Ihrer Bucket-Richtlinie verwenden, um alle Uploads zu verweigern, deren Nutzlasten nicht signiert sind. Beispielsweise: • Verweigern Sie Uploads, die den `Authorization` -Header zur Authentifizierung von Anfragen verwenden, aber die Nutzdaten nicht signieren. Weitere Informationen finden Sie unter Übertragen von Nutzdaten in einem einzelnen Datenblock in der Amazon Simple Storage Service API-Referenz. • Verweigern Sie Uploads, die vorsignierte URLs verwenden. Vorsignierte URLs haben immer eine `UNSIGNED_PAYLOAD` . Weitere Informationen finden Sie unter Authentifizierung von Anfragen und Authentifizierungsmethoden in der Amazon Simple Storage Service API-Referenz. Günstiger Wert: `UNSIGNED-PAYLOAD`	String

Wie API-Vorgänge autorisiert und authentifiziert werden

In der folgenden Tabelle sind Autorisierungs- und Authentifizierungsinformationen für S3-Express-One-Zone-API-Operationen aufgeführt werden. Für jeden API-Vorgang enthält die Tabelle den Namen der API-Operation, die IAM-Aktion, den Endpunkttyp (Regional oder Zonal) und den Autorisierungsmechanismus (IAM oder sitzungsbasiert). In dieser Tabelle wird auch angegeben, wo der kontoübergreifende Zugriff unterstützt wird. Der Zugriff auf Aktionen auf Bucket-Ebene kann nur über identitätsbasierte IAM-Richtlinien (Benutzer oder Rolle) und nicht über Bucket-Richtlinien gewährt werden.

API	Endpunkttyp	IAM-Aktion	Kontoübergreifender Zugriff
`CreateBucket`	Regional	`s3express:CreateBucket`	Nein
`DeleteBucket`	Regional	`s3express:DeleteBucket`	Nein
`ListDirectoryBuckets`	Regional	`s3express:ListAllMyDirectoryBuckets`	Nein
`PutBucketPolicy`	Regional	`s3express:PutBucketPolicy`	Nein
`GetBucketPolicy`	Regional	`s3express:GetBucketPolicy`	Nein
`DeleteBucketPolicy`	Regional	`s3express:DeleteBucketPolicy`	Nein
`CreateSession`	Zonal	`s3express:CreateSession`	Ja
`CopyObject`	Zonal	`s3express:CreateSession`	Ja
`DeleteObject`	Zonal	`s3express:CreateSession`	Ja
`DeleteObjects`	Zonal	`s3express:CreateSession`	Ja
`HeadObject`	Zonal	`s3express:CreateSession`	Ja
`PutObject`	Zonal	`s3express:CreateSession`	Ja

API	Endpunkttyp	IAM-Aktion	Kontoübergreifender Zugriff
`GetObjectAttributes`	Zonal	`s3express:CreateSession`	Ja
`ListObjectsV2`	Zonal	`s3express:CreateSession`	Ja
`HeadBucket`	Zonal	`s3express:CreateSession`	Ja
`CreateMultipartUpload`	Zonal	`s3express:CreateSession`	Ja
`UploadPart`	Zonal	`s3express:CreateSession`	Ja
`UploadPartCopy`	Zonal	`s3express:CreateSession`	Ja
`CompleteMultipartUpload`	Zonal	`s3express:CreateSession`	Ja
`AbortMultipartUpload`	Zonal	`s3express:CreateSession`	Ja
`ListParts`	Zonal	`s3express:CreateSession`	Ja
`ListMultipartUploads`	Zonal	`s3express:CreateSession`	Ja

Auf IAM-Identitäten basierende IAM-Richtlinien für S3 Express One Zone

Bevor Sie Verzeichnis-Buckets erstellen oder die Speicherklasse Amazon S3 Express One Zone verwenden können, müssen Sie Ihrer AWS Identity and Access Management (IAM)-Rolle oder Ihren Benutzern die erforderlichen Berechtigungen erteilen. Diese Beispielrichtlinie ermöglicht den Zugriff auf den `CreateSession`-API-Vorgang (zur Verwendung mit API-Vorgängen auf zonalem Endpunkt [Objektebene]) und auf alle API-Operationen des regionalen Endpunkts (Bucket-Ebene). Diese Richtlinie ermöglicht die Verwendung des `CreateSession`-API-Vorgangs mit allen Verzeichnis-

Buckets, aber die API-Operationen für regionale Endpunkte sind nur für die Verwendung mit dem angegebenen Verzeichnis-Bucket zulässig. Wenn Sie diese Beispielrichtlinie verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Sid": "AllowAccessRegionalEndpointAPIs",
             "Effect": "Allow",
             "Action": [
                 "s3express:DeleteBucket",
                 "s3express:DeleteBucketPolicy",
                 "s3express:CreateBucket",
                 "s3express:PutBucketPolicy",
                 "s3express:GetBucketPolicy",
                 "s3express:ListAllMyDirectoryBuckets"
             ],

             "Resource": "arn:aws:s3express:region:account_id:bucket/bucket-base-
name--azid--x-s3/*"
         },
         {
             "Sid": "AllowCreateSession",
             "Effect": "Allow",
             "Action": "s3express:CreateSession",
             "Resource": "*"
         }
     ]
 }
```

Beispiel für Verzeichnis-Bucket-Richtlinien für S3 Express One Zone

Dieser Abschnitt zeigt Verzeichnis-Bucket-Richtlinien zur Verwendung mit der Speicherklasse Amazon S3 Express One Zone. Wenn Sie diese Richtlinien verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre Informationen.

Die folgende Beispiel-Bucket-Richtlinie ermöglicht es AWS-Konto-ID *111122223333*, den `CreateSession`-API-Vorgang mit der `ReadWrite`-Standardsitzung für den angegebenen Verzeichnis-Bucket zu verwenden. Diese Richtlinie gewährt Zugriff auf die API-Operationen des zonalen Endpunkts (Objektebene).

Example – Bucket-Richtlinie, um **`CreateSession`**-Aufrufe mit der **`ReadWrite`**-Standardsitzung zuzulassen

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteAccess",
            "Effect": "Allow",
            "Resource": "arn:aws:s3express:us-west-2:account-id:bucket/bucket-base-name--azid--x-s3",
            "Principal": {
                "AWS": [
                    "111122223333"
                ]
            },
            "Action": [
                "s3express:CreateSession"
            ]
        }
    ]
}
```

Example – Bucket-Richtlinie, um **`CreateSession`**-Aufrufe mit einer **`ReadOnly`**-Sitzung zuzulassen

Das folgende Beispiel für eine Bucket-Richtlinie ermöglicht der AWS-Konto-ID *111122223333*, den `CreateSession`-API-Vorgang zu verwenden. Diese Richtlinie verwendet den `s3express:SessionMode`-Bedingungsschlüssel mit dem `ReadOnly`-Wert, um eine schreibgeschützte Sitzung einzurichten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadOnlyAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
```

```
            "Action": "s3express:CreateSession",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3express:SessionMode": "ReadOnly"
                }
            }
        }
    ]
}
```

Example – Bucket-Richtlinie, um den kontoübergreifenden Zugriff für **`CreateSession`**-Aufrufe zuzulassen

Die folgende Beispiel-Bucket-Richtlinie ermöglicht es AWS-Konto-ID *111122223333*, den `CreateSession`-API-Vorgang für den angegebenen Verzeichnis-Bucket zu verwenden, dessen Eigentümer AWS-Konto-ID *444455556666* ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccount",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": [
                "s3express:CreateSession"
            ],
            "Resource": "arn:aws:s3express:us-west-2:444455556666:bucket/bucket-base-name--azid--x-s3"

        }
    ]
}
```

`CreateSession`-Autorisierung

Amazon S3 Express One Zone unterstützt sowohl die AWS Identity and Access Management (AWS IAM)-Autorisierung als auch die sitzungsbasierte Autorisierung:

- Um regionale Endpunkt-API-Operationen (Operationen auf Bucket-Ebene oder Steuerebene) mit S3 Express One Zone zu verwenden, verwenden Sie das IAM-Autorisierungsmodell, das keine Sitzungsverwaltung beinhaltet. Berechtigungen werden für Aktionen einzeln erteilt. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.
- Um API-Vorgänge für zonale Endpunkte (Vorgänge auf Objektebene oder Datenebene) zu verwenden, verwenden Sie den `CreateSession`-API-Vorgang, um Sitzungen zu erstellen und zu verwalten, die für die Autorisierung von Datenanforderungen mit geringer Latenz optimiert sind. Um ein Sitzungstoken abzurufen und zu verwenden, müssen Sie die `s3express:CreateSession`-Aktion für Ihren Verzeichnis-Bucket in einer identitätsbasierten Richtlinie oder einer Bucket-Richtlinie zulassen. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone. Wenn Sie in der Amazon-S3-Konsole, über die AWS Command Line Interface (AWS CLI) oder mithilfe der AWS-SDKs auf S3 Express One Zone zugreifen, erstellt S3 Express One Zone für Sie eine Sitzung.

Wenn Sie die Amazon-S3-REST-API verwenden, können Sie dann den `CreateSession`API-Vorgang verwenden, um temporäre Sicherheitsanmeldeinformationen abzurufen, die eine Zugriffsschlüssel-ID, einen geheimen Zugriffsschlüssel, ein Sitzungstoken und eine Ablaufzeit enthalten. Die temporären Anmeldeinformationen bieten die gleichen Berechtigungen wie langfristige Sicherheitsanmeldeinformationen, z. B. IAM-Benutzer-Anmeldeinformationen müssen jedoch ein Sitzungstoken beinhalten.

Sitzungsmodus

Der Sitzungsmodus definiert den Umfang der Sitzung. In Ihrer Bucket-Richtlinie können Sie den `s3express:SessionMode`-Bedingungsschlüssel angeben, um zu steuern, wer eine `ReadWrite`- oder `ReadOnly`-Sitzung erstellen kann. Weitere Informationen zu `ReadWrite`- und `ReadOnly`-Sitzungen finden Sie unter dem `x-amz-create-session-mode`-Parameter für CreateSession in der Amazon-S3-API-Referenz. Informationen dazu, wie Sie eine Bucket-Richtlinie erstellen, finden Sie unter Beispiel für Verzeichnis-Bucket-Richtlinien für S3 Express One Zone.

Sitzungs-Token

Wenn Sie einen Aufruf mit temporären Sicherheitsanmeldeinformationen tätigen, muss dieser ein Sitzungs-Token enthalten. Das Sitzungstoken wird zusammen mit den temporären Anmeldeinformationen zurückgegeben. Ein Sitzungstoken ist auf Ihren Verzeichnis-Bucket beschränkt und wird verwendet, um zu überprüfen, ob die Sicherheitsanmeldedaten gültig und nicht abgelaufen sind. Um Ihre Sitzungen zu schützen, laufen temporäre Sicherheitsanmeldedaten nach 5 Minuten ab.

`CopyObject` und `HeadBucket`

Temporäre Sicherheitsanmeldedaten sind auf einen bestimmten Verzeichnis-Bucket beschränkt und werden automatisch für alle API-Aufrufe von zonalen Operationen (auf Objektebene) für einen bestimmten Verzeichnis-Bucket aktiviert. Im Gegensatz zu anderen API-Vorgängen für zonale Endpunkte verwenden `CopyObject`, `HeadBucket` und `CreateSession` keine Authentifizierung. Alle `CopyObject`- und `HeadBucket`-Anforderungen müssen mithilfe von IAM-Anmeldeinformationen authentifiziert und signiert werden. `CopyObject` und `HeadBucket` sind jedoch wie andere API-Vorgänge für zonale Endpunkte weiterhin von `s3express:CreateSession` autorisiert.

Weitere Informationen finden Sie unter CreateSession in der API-Referenz zu Amazon Simple Storage Service.

Bewährte Methoden für die Sicherheit in S3 Express One Zone

Amazon S3 Express One Zone enthält eine Reihe von Sicherheits-Features, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden stellen allgemeine Richtlinien und keine vollständige Sicherheitslösung dar. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Empfehlungen und nicht als bindend ansehen.

Standardeinstellungen für „Öffentlichen Zugriff blockieren“ und „Objekteigentümerschaft“

Um die Speicherklasse S3 Express One Zone zu verwenden, müssen Sie einen S3-Verzeichnis-Bucket verwenden. Verzeichnis-Buckets unterstützen S3 Block Public Access und S3 Object Ownership. Diese S3-Features werden für die Überwachung und Verwaltung des Zugriffs auf Ihre Buckets und Objekte verwendet.

Alle Einstellungen zum Blockieren jeglichen öffentlichen Zugriffs sind für Verzeichnis-Buckets standardmäßig aktiviert. Dazu ist die Objekt-Eigentümerschaft auf „Vom Bucket-Eigentümer

erzwungen“ festgelegt, was bedeutet, dass alle Zugriffssteuerungslisten (ACLs) deaktiviert sind. Diese Einstellungen können nicht geändert werden. Weitere Informationen zu diesen Features finden Sie unter the section called “Blockieren des öffentlichen Zugriffs” und the section called “Steuern der Objekteigentümerschaft”.

> **Note**
>
> Sie können keinen Zugriff auf Objekte gewähren, die in Verzeichnis-Buckets gespeichert sind. Sie können nur Zugriff auf Ihre Verzeichnis-Buckets gewähren. Das Autorisierungsmodell für S3 Express One Zone unterscheidet sich vom Autorisierungsmodell für Amazon S3. Weitere Informationen finden Sie unter `CreateSession`-Autorisierung.

Authentifizierung und Autorisierung

Die Authentifizierungs- und Autorisierungsmechanismen für S3 Express One Zone unterscheiden sich, je nachdem, ob Sie Anfragen für API-Operationen an zonalen Endpunkten oder API-Operationen an regionalen Endpunkten stellen. Zonale API-Operationen sind Operationen auf Objektebene (Datenebene). Regionale API-Operationen sind Operationen auf Bucket-Ebene (Steuerebene).

Mit S3 Express One Zone authentifizieren und autorisieren Sie Anforderungen an API-Operationen an zonalen Endpunkten mithilfe eines neuen sitzungsbasierten Mechanismus , der für die geringste Latenz optimiert ist. Bei der sitzungsbasierten Authentifizierung verwenden die AWS-SDKs den `CreateSession`-API-Vorgang , um temporäre Anmeldeinformationen anzufordern, die den Zugriff auf Ihren Verzeichnis-Bucket mit geringer Latenz ermöglichen. Diese temporären Anmeldeinformationen sind auf einen bestimmten Verzeichnis-Bucket beschränkt und laufen nach 5 Minuten ab. Sie können diese temporären Anmeldeinformationen verwenden, um zonale API-Aufrufe (Objektebene) zu signieren. Weitere Informationen finden Sie unter `CreateSession`-Autorisierung.

Signieren von Anforderungen mit S3-Express-One-Zone-Anmeldeinformationen

Sie verwenden Ihre S3-Express-One-Zone-Anmeldeinformationen, um API-Anfragen für zonale Endpunkte (Objektebene) mit AWS Signature Version 4 als `s3express`-Servicename zu signieren. Wenn Sie Ihre Anforderungen signieren, verwenden Sie den geheimen Schlüssel, der von `CreateSession`zurückgegeben wurde, und stellen auch das Sitzungstoken mit dem `x-amzn-s3session-token header` bereit. Weitere Informationen finden Sie unter CreateSession.

Die unterstützten AWS-SDKs für die Klasse S3 Express One Zone verwalten Anmeldeinformationen und Signaturen für Sie. Wir empfehlen, die AWS-SDKs für S3 Express One Zone zu verwenden, um Anmeldeinformationen zu aktualisieren und Anfragen für Sie zu signieren.

Signieren von Anforderungen mit IAM-Anmeldeinformationen

Alle regionalen API-Aufrufe (Bucket-Ebene) müssen authentifiziert und mit AWS Identity and Access Management (IAM)- Anmeldeinformationen statt mit temporären Sitzungsanmeldedaten signiert werden. IAM-Anmeldeinformationen bestehen aus der Zugriffsschlüssel-ID und dem geheimen Zugriffsschlüssel für die IAM-Identitäten. Alle `CopyObject`- und `HeadBucket`-Anforderungen müssen außerdem mithilfe von IAM-Anmeldeinformationen authentifiziert und signiert werden.

Um die geringste Latenz für Ihre zonalen Betriebsaufrufe (auf Objektebene) zu erreichen, empfehlen wir, zum Signieren Ihrer Anfragen die S3-Express-One-Zone-Anmeldeinformationen zu verwenden, die Sie beim Aufruf von `CreateSession` erhalten haben, mit Ausnahme von Anforderungen an `CopyObject` und `HeadBucket`.

Verwenden von AWS CloudTrail

AWS CloudTrail bietet Aufzeichnungen der Aktionen eines Benutzers, einer Rolle oder eines AWS-Service in Amazon S3. Sie können die von gesammelten Informationen verwenden CloudTrail , um Folgendes zu bestimmen:

- Die Anforderung, die an Amazon S3 gestellt wurde
- Die IP-Adresse, von der die Anforderung erfolgt ist
- Wer die Anforderung gestellt hat
- Wann die Anforderung gestellt wurde
- Zusätzliche Details zur Anforderung

Wenn Sie Ihr einrichtenAWS-Konto, CloudTrail ist standardmäßig aktiviert. Die folgenden regionalen Endpunkt-API-Operationen (API-Operationen auf Bucket-Ebene oder Steuerebene) werden in protokolliert CloudTrail.

- `CreateBucket`
- `DeleteBucket`
- `DeleteBucketPolicy`
- `PutBucketPolicy`

- `GetBucketPolicy`
- `ListDirectoryBuckets`

Sie können aktuelle Ereignisse in der - CloudTrail Konsole anzeigen. Um eine fortlaufende Aufzeichnung der Aktivitäten und Ereignisse für Ihre Amazon S3-Buckets zu erstellen, können Sie einen Trail in der CloudTrail Konsole erstellen. Weitere Informationen finden Sie unter Creating a trail im AWS CloudTrail-Benutzerhandbuch.

> **Note**
>
> Für S3 Express One Zone wird CloudTrail die Protokollierung von API-Operationen für zonale Endpunkte (Objektebene oder Datenebene) (z. B. `PutObject` oder `GetObject`) nicht unterstützt.

Implementieren der Überwachung mit AWS-Überwachungstools

Die Überwachung ist ein wichtiger Teil der Aufrechterhaltung der Zuverlässigkeit, Sicherheit, Verfügbarkeit und Leistung von Amazon S3 und Ihrer AWS-Lösungen. AWS bietet verschiedene Tools und Services, die Sie bei der Überwachung von Amazon S3 und Ihrer anderen AWS-Services unterstützen. Sie können beispielsweise Amazon- CloudWatch Metriken für Amazon S3 überwachen, insbesondere die `NumberOfObjects` Speichermetriken `BucketSizeBytes` und .

Objekte, die in der Speicherklasse S3 Express One Zone gespeichert sind, werden in den Speichermetriken `BucketSizeBytes` und `NumberOfObjects` für Amazon S3 nicht berücksichtigt. Die Speichermetriken `BucketSizeBytes` und `NumberOfObjects` werden jedoch für S3 Express One Zone unterstützt. Um die Metriken Ihrer Wahl zu sehen, können Sie zwischen den Amazon-S3-Speicherklassen und der Speicherklasse S3 Express One Zone unterscheiden, indem Sie eine `StorageType`-Dimension angeben. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch und Überwachen von Amazon S3.

Optimieren der Leistung von Amazon S3 Express One Zone

Amazon S3 Express One Zone ist eine leistungsstarke S3-Speicherklasse mit einer einzigen Availability Zone (AZ), die speziell für den konsistenten Datenzugriff im einstelligen

Millisekundenbereich für Ihre latenzempfindlichsten Anwendungen entwickelt wurde. S3 Express One Zone ist die erste S3-Speicherklasse, mit der Sie leistungsstarke Objektspeicher- und AWS-Computingressourcen wie Amazon Elastic Compute Cloud, Amazon Elastic Kubernetes Service und Amazon Elastic Container Service innerhalb einer einzigen Availability Zone zusammenführen können. Die gemeinsame Nutzung Ihrer Speicher- und Computingressourcen optimiert die Rechenleistung und die Kosten und sorgt für eine höhere Datenverarbeitungsgeschwindigkeit.

S3 Express One Zone bietet eine ähnliche Leistungselastizität wie andere S3-Speicherklassen, jedoch mit konsistenten Latenzen für Lese- und Schreibanfragen im ersten Byte im einstelligen Millisekundenbereich – bis zu zehnmal schneller als S3 Standard. S3 Express One Zone wurde von Grund auf für die Unterstützung von Burst-Durchsätzen bis hin zu sehr hohen Aggregatwerten konzipiert. Die Speicherklasse S3 Express One Zone verwendet eine maßgeschneiderte Architektur, um die Leistung zu optimieren und durch die Speicherung von Daten auf Hochleistungshardware eine konstant niedrige Anforderungslatenz zu gewährleisten. Das Objektprotokoll für S3 Express One Zone wurde verbessert, um die Authentifizierung und den Metadaten-Overhead zu optimieren.

Um die Zugriffsgeschwindigkeit weiter zu erhöhen und Hunderttausende von Anfragen pro Sekunde zu unterstützen, speichert S3 Express One Zone Daten außerdem in einem neuen Bucket-Typ, einem Amazon-S3-Verzeichnis-Bucket. Jeder S3-Verzeichnis-Bucket kann Hunderttausende von Transaktionen pro Sekunde (TPS) unterstützen.

Die Kombination aus leistungsstarker, speziell entwickelter Hardware und Software, die Datenzugriffsgeschwindigkeiten im einstelligen Millisekundenbereich bietet, und Verzeichnis-Buckets, die für eine große Anzahl von Transaktionen pro Sekunde skalierbar sind, machen S3 Express One Zone zur besten Amazon-S3-Speicherklasse für anforderungsintensive Operationen oder leistungskritische Anwendungen.

Die folgenden Themen beschreiben bewährte Verfahren und Designmuster zur Optimierung der Leistung von Anwendungen, die die Speicherklasse Sr Express One Zone verwenden.

Themen

Leistungsrichtlinien und Entwurfsmuster für S3 Express One Zone

Bei der Entwicklung von Anwendungen, die Objekte zu Amazon S3 Express One Zone hochladen und daraus abrufen, sollten Sie unsere bewährten Methoden befolgen, um die Leistung zu optimieren. Um die Speicherklasse S3 Express One Zone zu verwenden, müssen Sie einen S3-

Verzeichnis-Bucket erstellen. Die Speicherklasse S3 Express One Zone wird für die Verwendung mit S3-Allzweck-Buckets nicht unterstützt.

Leistungsrichtlinien für alle anderen Amazon-S3-Speicherklassen und S3-Allzweck-Buckets finden Sie unter Bewährte Methoden für Designmuster: Optimieren der Leistung von Amazon S3.

Zur Erzielung der besten Leistung für Ihre Anwendung bei Verwendung der Speicherklasse S3 Express One Zone und von Verzeichnis-Buckets empfehlen wir die folgenden Vorgehensweisen und Designmuster.

Themen

Platzieren Sie den S3-Express-One-Zone-Speicher gemeinsam mit Ihren AWS-Computingressourcen

Jeder Verzeichnis-Bucket wird in einer einzigen Availability Zone gespeichert, die Sie bei dessen Erstellung auswählen. Sie können damit beginnen, indem Sie neuen Verzeichnis-Bucket in einer Availability Zone erstellen, die für Ihre Computng-Workloads oder Ressourcen lokal ist. Sie können dann sofort mit Lese- und Schreibvorgängen mit sehr niedriger Latenz beginnen. Verzeichnis-Buckets sind die ersten S3-Buckets, in denen Sie die Availability Zone in einer AWS-Region auswählen können, um die Latenz zwischen Rechenleistung und Speicher zu reduzieren.

Wenn Sie in verschiedenen Availability Zones auf Verzeichnis-Buckets zugreifen, erhöht sich die Latenz. Um die Leistung zu optimieren, empfehlen wir, dass Sie von Amazon Elastic Container Service-, Amazon Elastic Kubernetes Service- und Amazon Elastic Compute Cloud-Instances, die sich nach Möglichkeit in derselben Availability Zone befinden, auf einen Directory-Bucket zugreifen.

Verzeichnis-Buckets

Jeder Verzeichnis-Bucket kann Hunderttausende von Transaktionen pro Sekunde (TPS) unterstützen. Im Gegensatz zu Allzweck-Buckets organisieren Verzeichnis-Buckets Schlüssel hierarchisch in Verzeichnissen statt nach Präfixen. Ein Präfix ist eine Zeichenfolge am Anfang des Objektschlüsselnamens. Sie können sich Präfixe als eine Möglichkeit vorstellen, Ihre Daten ähnlich wie Verzeichnisse zu organisieren. Präfixe sind jedoch keine Verzeichnisse.

Präfixe organisieren Daten in einem flachen Namespace innerhalb von Allzweck-Buckets und die Anzahl der Präfixe in einem Allzweck-Bucket ist unbegrenzt. Jedes Präfix kann mindestens 3 500 `PUT`/`POST`/`DELETE` oder 5 500 `GET`/`HEAD`-Anforderungen pro Sekunde erreichen. Sie können Anfragen auch über mehrere Präfixe parallelisieren, um die Leistung zu skalieren. Die Skalierung erfolgt sowohl bei Lese- als auch bei Schreiboperationen schrittweise und nicht sofort. Während Allzweck-Buckets auf Ihre neue höhere Anforderungsrate skaliert werden, erhalten Sie möglicherweise einige HTTP-Statuscode 503 (Service Unavailable)-Fehler.

Bei einem hierarchischen Namespace ist das Trennzeichen im Objektschlüssel wichtig. Das einzige unterstützte Trennzeichen ist der Schrägstrich (`/`). Verzeichnisse werden durch Trennzeichengrenzen bestimmt. Beispielsweise führt der Objektschlüssel `dir1/dir2/file1.txt` dazu, dass die Verzeichnisse `dir1/`und `dir2/` automatisch erstellt werden und das Objekt `file1.txt` dem `/dir2`-Verzeichnis im Pfad `dir1/dir2/file1.txt` hinzugefügt wird.

Die Verzeichnisse, die erstellt werden, wenn Objekte in Verzeichnis-Buckets hochgeladen werden, haben keine TPS-Beschränkungen pro Präfix und werden automatisch vorskaliert, um die Wahrscheinlichkeit von HTTP 503-Fehlern (Service Unavailable) zu verringern. Diese automatische Skalierung ermöglicht Ihren Anwendungen, Lese- und Schreibanforderungen innerhalb von Verzeichnissen und zwischen Verzeichnissen nach Bedarf zu parallelisieren.

Parallelisierung horizontaler Skalierungsanforderungen für verzeichnis-Buckets

Sie erreichen die beste Leistung durch die Ausgabe mehrerer gleichzeitiger Anfragen an Verzeichnis-Buckets zur Verteilung Ihrer Anforderungen auf separate Verbindungen und zur Maximierung der verfügbaren Bandbreite. S3Express One Zone hat keine Beschränkungen für die Anzahl der Verbindungen, die mit Ihrem Verzeichnis-Bucket hergestellt werden. Einzelne Verzeichnisse können die Leistung horizontal und automatisch skalieren, wenn eine große Anzahl gleichzeitiger Schreibvorgänge in dasselbe Verzeichnis stattfindet.

Wenn ein Objektschlüssel zum ersten Mal erstellt wird und sein Schlüsselname ein Verzeichnis enthält, wird das Verzeichnis automatisch für das Objekt erstellt. Bei nachfolgenden Objekt-Uploads

in dasselbe Verzeichnis muss das Verzeichnis nicht erstellt werden, wodurch die Latenz beim Hochladen von Objekten in bestehende Verzeichnisse reduziert wird.

Obwohl sowohl flache als auch tiefe Verzeichnisstrukturen für das Speichern von Objekten in einem Verzeichnis-Bucket unterstützt werden, werden Verzeichnis-Buckets automatisch horizontal skaliert, wodurch die Latenz bei gleichzeitigen Uploads in dasselbe Verzeichnis oder in parallele „Verzeichnisgeschwister“ geringer ist.

Verwenden der sitzungsbasierten Authentifizierung

S3 Express One Zone und Verzeichnis-Buckets unterstützen einen neuen sitzungsbasierten Autorisierungsmechanismus zur Authentifizierung und Autorisierung von Anforderungen an einen Verzeichnis-Bucket. Bei der sitzungsbasierten Authentifizierung verwenden die AWS-SDKs automatisch den `CreateSession`-API-Vorgang, um ein temporäres Sitzungstoken zu erstellen, das für die Autorisierung von Datenanforderungen an einen Verzeichnis-Bucket mit geringer Latenz verwendet werden kann.

Die AWS-SDKs verwenden den `CreateSession`-API-Vorgang, um temporäre Anmeldeinformationen anzufordern. Anschließend erstellen und aktualisieren sie für Sie automatisch alle 5 Minuten Token. Um die Leistungsvorteile der Speicherklasse S3 Express One Zone zu nutzen, wird empfohlen, die AWS-SDKs zum Initiieren und Verwalten der `CreateSession`-API-Anforderung zu verwenden. Weitere Informationen zu diesem sitzungsbasierten Modell finden Sie unter `CreateSession`-Autorisierung.

Bewährte Methoden für zusätzliche S3-Prüfsummen

S3 Express One Zone bietet Ihnen die Möglichkeit, den Prüfsummenalgorithmus auszuwählen, der zur Validierung Ihrer Daten beim Hoch- oder Herunterladen verwendet wird. Sie können einen der folgenden Secure Hash Algorithms (SHA)- oder Cyclic Redundancy Check (CRC)-Algorithmen zur Überprüfung der Datenintegrität auswählen: CRC32, CRC32C, SHA-1 und SHA-256. MD5-based Prüfsummen werden von der Speicherklasse S3 Express One Zone nicht unterstützt.

CRC32 ist die Standardprüfsumme, die von den AWS-SDKs bei der Übertragung von Daten zu oder von S3 Express One Zone verwendet wird. Wir empfehlen die Verwendung von CRC32 und CRC32C, um die beste Leistung mit der Speicherklasse S3 Express One Zone zu erzielen.

Verwenden der neuesten Version der AWS-SDKs und der gemeinsamen Laufzeitbibliotheken

Einige der AWS-SDKs stellen auch die AWS Common Runtime (CRT)-Bibliotheken bereit, um die Leistung in S3-Clients weiter zu beschleunigen. Zu diesen SDKs gehören AWS SDK for Java 2.x, AWS SDK for C++ und AWS SDK for Python (Boto3). Der CRT-basierte S3-Client überträgt Objekte zu und von S3 Express One Zone mit verbesserter Leistung und Zuverlässigkeit, indem er automatisch den mehrteiligen Upload-API-Vorgang und Abrufe im Bytebereich verwendet, um horizontal skalierende Verbindungen zu automatisieren.

Um die höchste Leistung mit der Speicherklasse S3 Express One Zone zu erzielen, empfehlen wir, die neueste Version der AWS-SDKs zu verwenden, die die CRT-Bibliotheken enthalten, oder die AWS Command Line Interface (AWS CLI) zu verwenden.

Fehlerbehebung bei der Leistung

Wiederholung von Anforderungen für latenzsensitive Anwendungen

S3 Express One Zone wurde speziell für gleichbleibende Leistung ohne zusätzliche Anpassungen entwickelt. Die Festlegung aggressiver Timeout-Werte und Wiederholungsversuche kann jedoch weiter zu einer gleichbleibenden Latenz und Leistung beitragen. Die AWS-SDKs verfügen über konfigurierbare Timeout- und Wiederholungsversuch-Werte, die Sie an die Toleranzen Ihrer spezifischen Anwendung anpassen können.

AWS Common Runtime (CRT)-Bibliotheken und Kopplung von Amazon-EC2-Instance-Typen

Anwendungen, die eine große Anzahl von Lese- und Schreibvorgängen ausführen, benötigen wahrscheinlich mehr Arbeitsspeicher oder Computing-Kapazitäten als Anwendungen, bei denen dies nicht der Fall ist. Wählen Sie beim Starten Ihrer Amazon-Elastic-Compute-Cloud (Amazon EC2)-Instances für Ihre leistungsforderndne Workloads die Instance-Typen aus, die über die Menge dieser Ressourcen verfügen, die Ihre Anwendung benötigt. Die S3-Express-One-Zone-Hochleistungsspeicherung lässt sich ideal mit größeren und neueren Instance-Typen mit größerem Systemspeicher und leistungsstärkeren CPUs und GPUs kombinieren, die die Vorteile von leistungsfähigerem Speicher nutzen können. Wir empfehlen außerdem, die neuesten Versionen der CRT-fähigen AWS-SDKs zu verwenden, die Lese- und Schreibanforderungen besser parallel beschleunigen können.

Verwenden Sie die sitzungsbasierte Authentifizierung in AWS-SDKs anstelle der HTTP-REST-APIs

Mit Amazon S3 können Sie auch die Leistung optimieren, wenn Sie HTTP-REST-API-Anfragen verwenden, indem Sie dieselben bewährten Methoden befolgen, die Teil der AWS-SDKs sind. Angesichts des sitzungsbasierten Autorisierungs- und Authentifizierungsmechanismus, der von S3 Express One Zone verwendet wird, empfehlen wir Ihnen jedoch dringend, die AWS-SDKs zur

Verwaltung von `CreateSession` und das zugehörige verwaltete Sitzungstoken zu verwenden. Die AWS-SDKs erstellen und aktualisieren mithilfe der `CreateSession`-API-Operation automatisch Token in Ihrem Namen. Die Verwendung von `CreateSession` ermöglicht Einsparungen bei der Round-Trip-Latenz pro Anforderung an AWS Identity and Access Management (IAM) für die Autorisierung der einzelnen Anforderungen.

Entwicklung mit S3 Express One Zone

Amazon S3 Express One Zone ist die erste S3-Speicherklasse, bei der Sie eine einzelne Availability Zone mit der Option auswählen können, Ihren Objektspeicher zusammen mit Ihren Computingressourcen zu platzieren, was die höchstmögliche Zugriffsgeschwindigkeit bietet. Mit der Speicherklasse S3 Express One Zone verwenden Sie S3-Verzeichnis-Buckets zum Speichern Ihrer Daten. Jeder Verzeichnis-Bucket verwendet die Speicherklasse S3 Express One Zone, um Objekte in einer einzigen Availability Zone zu speichern, die Sie bei der Erstellung des Buckets auswählen können.

Nachdem Sie Ihren Verzeichnis-Bucket erstellt haben, können Sie sofort mit Lese- und Schreibvorgängen mit sehr niedriger Latenz beginnen. Sie können mit Ihrem Verzeichnis-Bucket mit einer Endpunktverbindung über eine Virtual Private Cloud (VPC) kommunizieren oder Sie können zonale und regionale API-Vorgänge verwenden, um Ihre Objekte und Verzeichnis-Buckets zu verwalten. Sie können die Speicherklasse S3 Express One Zone auch über die Amazon-S3-Konsole, AWS Command Line Interface (AWS CLI), AWS-SDKs und die Amazon-S3-REST-API verwenden.

Die Speicherklasse Amazon S3 Express One Zone ist für eine Verfügbarkeit von 99,95 Prozent innerhalb einer einzigen Availability Zone konzipiert und wird durch das Amazon S3 Service Level Agreement unterstützt. Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. S3 Express One Zone wurde entwickelt, um mit gleichzeitigen Geräteausfällen umzugehen, indem verlorene Redundanz schnell erkannt und behoben werden kann. Wenn das vorhandene Gerät ausfällt, leitet S3 Express One Zone Anfragen automatisch an neue Geräte innerhalb einer Availability Zone weiter. Diese Redundanz trägt dazu bei, den unterbrechungsfreien Zugriff auf Ihre Daten innerhalb einer Availability Zone sicherzustellen.

Themen

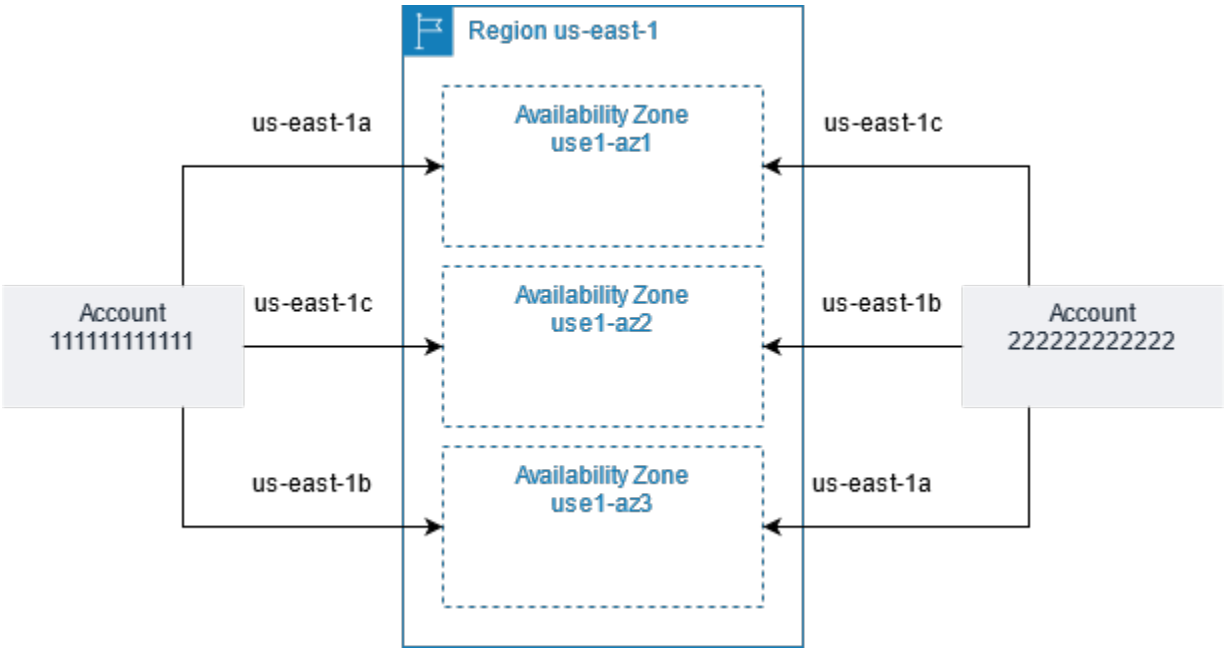
- Availability Zones und Regionen bei S3 Express One Zone
- Regionale und zonale Endpunkte
- S3-Express-One-Zone-API-Operationen

Availability Zones und Regionen bei S3 Express One Zone

Eine Availability Zone ist eines oder mehrere diskrete Rechenzentren mit redundanter Stromversorgung, Vernetzung und Konnektivität in einem AWS-Region. Um Abrufe mit niedriger Latenz zu optimieren, werden Objekte in der Speicherklasse Amazon S3 Express One Zone redundant in S3-Verzeichnis-Buckets in einer einzigen Availability Zone gespeichert, die für Ihr Computing-Workload lokal ist. Wenn Sie einen Verzeichnis-Bucket erstellen, wählen Sie die Availability Zone und die AWS-Region für Ihren Bucket.

AWS ordnet die physischen Availability Zones nach dem Zufallsprinzip den jeweiligen Availability-Zone-Namen für jedes AWS-Konto zu. Dieser Ansatz hilft dabei, Ressourcen auf die Availability Zones in einer AWS-Region zu verteilen, anstatt dass sich die Ressourcen wahrscheinlich auf die erste Availability Zone für jede Region konzentrieren. Daher entspricht die Availability Zone `us-east-1a` für Ihr AWS-Konto möglicherweise nicht demselben physischen Standort wie `us-east-1a` für ein anderes AWS-Konto. Weitere Informationen finden Sie unter Regionen und Availability Zones im Benutzerhandbuch zu Amazon EC2 für Linux-Instances.

Um die Availability Zones kontenübergreifend zu koordinieren, müssen Sie die AZ-ID verwenden, die eine eindeutige und konsistente Kennung für eine Availability Zone ist. Beispielsweise ist `use1-az1` eine AZ-ID für die `us-east-1`-Region und hat in jedem AWS-Konto den gleichen physischen Standort. Die folgende Abbildung zeigt, dass die AZ-IDs für jedes Konto identisch sind, auch wenn die Namen der Availability Zone für jedes Konto unterschiedlich zugeordnet werden können.



Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. S3 Express One Zone ist für eine Verfügbarkeit von 99,95 Prozent innerhalb einer einzigen Availability Zone konzipiert und wird durch das Amazon S3 Service Level Agreement unterstützt. Weitere Informationen finden Sie unter Einzelne Availability Zone.

S3 Express One Zone wird in den folgenden Regionen und Availability Zones unterstützt:

Von S3 Express One Zone unterstützte Regionen und Availability Zones

Name der Region	Regionscode	Availability Zone-ID
USA Ost (Nord-Virginia)	`us-east-1`	`use1-az4`
		`use1-az5`
		`use1-az6`
USA West (Oregon)	`us-west-2`	`usw2-az1`
		`usw2-az3`
		`usw2-az4`
Asien-Pazifik (Tokio)	`ap-northeast-1`	`apne1-az1`
		`apne1-az4`
Europa (Stockholm)	`eu-north-1`	`eun1-az1`
		`eun1-az2`
		`eun1-az3`

Regionale und zonale Endpunkte

Um von Ihrer Virtual Private Cloud (VPC) aus auf die regionalen und zonalen Endpunkte für Amazon S3 Express One Zone zuzugreifen, können Sie Gateway-VPC-Endpunkte verwenden. Nachdem Sie den Gateway-Endpunkt erstellt haben, können Sie ihn als Ziel in Ihrer Routing-Tabelle für Datenverkehr hinzufügen, der von Ihrer VPC zu S3 Express One Zone bestimmt ist. Für die Nutzung

von Gateway-Endpunkten fallen keine zusätzlichen Gebühren an. Weitere Informationen dazu, wie Sie Gateway-VPC-Endpunkte konfigurieren, finden Sie unter Networking für S3 Express One Zone.

Wenn Sie mit S3 Express One Zone arbeiten, sind API-Vorgänge auf Bucket-Ebene (Steuerebene) über einen regionalen Endpunkt verfügbar und werden als API-Vorgänge für regionale Endpunkte bezeichnet. Beispiele für API-Operationen für regionalen Endpunkte sind `CreateBucket` und `DeleteBucket`.

Nachdem Sie einen Verzeichnis-Bucket erstellt haben, können Sie zonale (Objektebene- oder Datenebenen-Endpunkt-API-Operationen) verwenden, um die Objekte in Ihrem Verzeichnis-Bucket hochzuladen und zu verwalten. API-Vorgänge für zonale Endpunkte sind über einen zonalen Endpunkt verfügbar. Beispiele für zonale API-Operationen sind `PutObject` und `CopyObject`.

S3-Express-One-Zone-API-Operationen

Die Speicherklasse Amazon S3 Express One Zone unterstützt sowohl regionale (Bucket-Ebene oder Steuerebene) als auch zonale (Objektebene oder Datenebene) Endpunkt-API-Operationen. Weitere Informationen finden Sie unter Networking für S3 Express One Zone und Endpunkte und Gateway-VPC-Endpunkte.

Regionale Endpunkt-API-Operationen

Die folgenden regionalen Endpunkt-API-Operationen werden für S3 Express One Zone unterstützt:

- CreateBucket
- DeleteBucket
- DeleteBucketPolicy
- GetBucketPolicy
- ListDirectoryBuckets
- PutBucketPolicy

Zonale Endpunkt-API-Operationen

Die folgenden zonalen Endpunkt-API-Operationen werden für S3 Express One Zone unterstützt:

- CreateSession
- CopyObject
- DeleteObject

- DeleteObjects
- GetObject
- GetObjectAttributes
- HeadBucket
- HeadObject
- ListObjectsV2
- PutObject
- AbortMultipartUpload
- CompleteMultiPartUpload
- CreateMultipartUpload
- ListMultipartUploads
- ListParts
- UploadPart
- UploadPartCopy

Verwalten des Datenzugriffs mit Amazon S3-Zugangspunkten

Amazon S3-Zugriffspunkte vereinfachen den Datenzugriff für jeden AWS Service oder jede Kundenanwendung, die Daten in S3 speichert. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Jeder Zugriffspunkt verfügt über unterschiedliche Berechtigungen und Netzwerkkontrollen, die S3 für alle Anforderungen anwendet, die über diesen Zugriffspunkt eingehen. Jeder Zugriffspunkt erzwingt eine benutzerdefinierte Zugriffspunktrichtlinie, die in Verbindung mit der Bucket-Richtlinie funktioniert, die dem zugrunde liegenden Bucket zugeordnet ist. Sie können jeden Zugriffspunkt so konfigurieren, dass nur Anforderungen aus einer Virtual Private Cloud (VPC) akzeptiert werden, um den Amazon S3-Datenzugriff auf ein privates Netzwerk zu beschränken. Sie können auch benutzerdefinierte Block Public Access-Einstellungen für jeden Zugriffspunkt konfigurieren.

> **Note**
>
> - Sie können Zugriffspunkte nur zum Ausführen von Vorgängen an Objekten verwenden. Sie können mit Zugriffspunkten keine anderen Amazon S3-Vorgänge auszuführen, z. B. das Ändern oder Löschen von Buckets. Eine vollständige Liste der S3-Vorgänge, die Zugriffspunkte unterstützen, finden Sie unter Kompatibilität von Zugriffspunkten mit - AWS Services.
> - Zugriffspunkte funktionieren mit einigen, aber nicht allen AWS Services und Funktionen. Beispielsweise können Sie die regionsübergreifende Replikation nicht so konfigurieren, dass sie über einen Zugriffspunkt ausgeführt wird. Eine vollständige Liste der AWS Services, die mit S3-Zugriffspunkten kompatibel sind, finden Sie unter Kompatibilität von Zugriffspunkten mit - AWS Services.

In diesem Abschnitt wird erläutert, wie Sie mit Amazon S3-Zugriffspunkten arbeiten. Hinweise zum Arbeiten mit Buckets finden Sie unter Bucket-Übersicht. Weitere Informationen zur Arbeit mit Objekten finden Sie unter Übersicht über Amazon-S3-Objekte.

Themen

- Erstellen von Zugriffspunkten
- Verwenden von Zugriffspunkten
- Einschränkungen von Access Points

Konfigurieren von IAM-Richtlinien für die Verwendung von Zugriffspunkten

Amazon S3-Zugriffspunkte unterstützen AWS Identity and Access Management (IAM)-Ressourcenrichtlinien, mit denen Sie die Verwendung des Zugriffspunkts nach Ressource, Benutzer oder anderen Bedingungen steuern können. Damit eine Anwendung oder ein Benutzer über einen Zugriffspunkt auf Objekte zugreifen kann, müssen sowohl der Zugriffspunkt als auch der zugrunde liegende Bucket die Anforderung zulassen.

> **Important**
>
> Das Hinzufügen eines S3-Zugriffspunkts zu einem Bucket ändert nicht das Verhalten des Buckets, wenn über den vorhandenen Bucket-Namen oder den Amazon-Ressourcennamen (ARN) auf ihn zugegriffen wird. Alle vorhandenen Vorgänge für den Bucket funktionieren weiterhin wie zuvor. Einschränkungen, die Sie in eine Zugriffspunktrichtlinie einschließen, gelten nur für Anforderungen, die über diesen Zugriffspunkt eingehen.

Wenn Sie IAM-Ressourcenrichtlinien verwenden, stellen Sie sicher, dass Sie Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge von beheben, AWS Identity and Access Management Access Analyzer bevor Sie Ihre Richtlinie speichern. IAM Access Analyzer führt Richtlinienprüfungen durch, um Ihre Richtlinie anhand der IAM-Richtliniengrammatik und der bewährten Methoden zu validieren. Diese Prüfungen generieren Ergebnisse und bieten Empfehlungen, die Sie beim Erstellen von Richtlinien unterstützen, die funktionsfähig sind und den bewährten Methoden für Sicherheit entsprechen.

Weitere Informationen zum Validieren von Richtlinien mit IAM Access Analyzer finden Sie unter Validierung der IAM-Access-Analyzer-Richtlinien im IAM-Benutzerhandbuch. Eine Liste der Warnungen, Fehler und Vorschläge, die von IAM Access Analyzer zurückgegeben werden, finden Sie unter IAM-Access-Analyzer-Richtlinienprüfungsreferenz.

Beispiele von -Zugriffspunktrichtlinien

In den folgenden Beispielen wird veranschaulicht, wie IAM-Richtlinien zum Steuern von Anforderungen erstellt werden, die über einen Zugriffspunkt eingehen.

> **Note**
>
> Berechtigungen, die in einer Zugriffspunktrichtlinie erteilt werden, sind nur wirksam, wenn der zugrunde liegende Bucket auch denselben Zugriff zulässt. Sie können dies auf zwei Arten erreichen:
>
> 1. (Empfohlen) Delegieren Sie die Zugriffssteuerung vom Bucket an den Zugriffspunkt, wie unter Delegieren der Zugangskontrolle an Zugriffspunkte beschrieben.
> 2. Fügen Sie der Richtlinie des zugrunde liegenden Buckets dieselben Berechtigungen hinzu, die in der Zugriffspunktrichtlinie enthalten sind. Beispiel 1 für eine Zugriffspunktrichtlinie veranschaulicht, wie die zugrunde liegende Bucket-Richtlinie geändert wird, damit der erforderliche Zugriff gewährt wird.

Example 1 – Erteilung der Zugriffspunktberechtigung

Die folgende Zugriffspunktrichtlinie gewährt IAM-Benutzer *Jane* in Konto *123456789012* Berechtigungen für GET- und PUT-Objekte mit dem Präfix *Jane/* über Zugriffspunkt *my-access-point* in Konto *123456789012*.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Jane"
        },
        "Action": ["s3:GetObject", "s3:PutObject"],
        "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point/
object/Jane/*"
    }]
}
```

> **Note**
>
> Damit die Zugriffspunkt-Richtlinie effektiv Zugriff auf *Jane* gewährt, muss der zugrunde liegende Bucket *Jane* auch denselben Zugriff gewähren. Sie können die Zugriffssteuerung vom Bucket an den Zugriffspunkt delegieren, wie unter Delegieren der Zugangskontrolle an Zugriffspunkte beschrieben. Oder Sie können dem zugrunde liegenden Bucket die folgende Richtlinie hinzufügen, um Jane die erforderlichen Berechtigungen zu erteilen. Beachten Sie, dass sich der `Resource`-Eintrag zwischen den Zugriffspunkt- und Bucket-Richtlinien unterscheidet.
>
> ```
> {
>     "Version": "2012-10-17",
>     "Statement": [
>     {
>         "Effect": "Allow",
>         "Principal": {
>             "AWS": "arn:aws:iam::123456789012:user/Jane"
>         },
>         "Action": ["s3:GetObject", "s3:PutObject"],
>         "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/Jane/*"
>     }]
> }
> ```

Example 2 – Zugriffspunktrichtlinie mit Tag-Bedingung

Die folgende Zugriffspunktrichtlinie gewährt IAM-Benutzer *Mateo* in Konto *123456789012* Berechtigungen für GET-Objekte über Zugriffspunkt *my-access-point* in Konto *123456789012*, für deren Tag-Schlüssel *data* der Wert *finance* festgelegt ist.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect":"Allow",
        "Principal" : {
            "AWS": "arn:aws:iam::123456789012:user/Mateo"
        },
        "Action":"s3:GetObject",
```

```
        "Resource" : "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point/
object/*",
        "Condition" : {
            "StringEquals": {
                "s3:ExistingObjectTag/data": "finance"
            }
        }
    }]
}
```

Example 3 – Zugriffspunktrichtlinie, die eine Bucket-Auflistung ermöglicht

Die folgende Zugriffspunktrichtlinie berechtigt den IAM-Benutzer `Arnav` im Konto *123456789012* dazu, die Objekte in dem Bucket anzuzeigen, der Zugriffspunkt *my-access-point* im Konto *123456789012* zugrunde liegt.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Arnav"
        },
        "Action": "s3:ListBucket",
        "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point"
    }]
}
```

Example 4 – Service-Kontrollrichtlinie

Die folgende Service-Kontrollrichtlinie erfordert, dass alle neuen Zugriffspunkte mit einem Virtual Private Cloud (VPC)-Netzwerkursprung erstellt werden. Mit dieser Richtlinie können Benutzer in Ihrer Organisation keine neuen Zugriffspunkte erstellen, auf die über das Internet zugegriffen werden kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Deny",
        "Action": "s3:CreateAccessPoint",
        "Resource": "*",
```

```
        "Condition": {
            "StringNotEquals": {
                "s3:AccessPointNetworkOrigin": "VPC"
            }
        }
    }]
}
```

Example 5 – Bucket-Richtlinie zur Begrenzung von S3-Vorgängen für VPC-Netzwerkursprünge

Die folgende Bucket-Richtlinie beschränkt den Zugriff auf alle S3-Objekt-Vorgänge für Bucket *DOC-EXAMPLE-BUCKET* auf Zugriffspunkte mit einem VPC-Netzwerkursprung.

> **Important**
>
> Bevor Sie eine Anweisung wie in diesem Beispiel gezeigt verwenden, stellen Sie sicher, dass Sie keine Funktionen verwenden müssen, die von Zugriffspunkten nicht unterstützt werden, z. B. regionsübergreifende Replikation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:BypassGovernanceRetention",
                "s3:DeleteObject",
                "s3:DeleteObjectTagging",
                "s3:DeleteObjectVersion",
                "s3:DeleteObjectVersionTagging",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectLegalHold",
                "s3:GetObjectRetention",
                "s3:GetObjectTagging",
                "s3:GetObjectVersion",
                "s3:GetObjectVersionAcl",
                "s3:GetObjectVersionTagging",
                "s3:ListMultipartUploadParts",
```

```
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectLegalHold",
                "s3:PutObjectRetention",
                "s3:PutObjectTagging",
                "s3:PutObjectVersionAcl",
                "s3:PutObjectVersionTagging",
                "s3:RestoreObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:AccessPointNetworkOrigin": "VPC"
                }
            }
        }
    ]
}
```

Bedingungsschlüssel

S3-Zugriffspunkte verwenden Bedingungsschlüssel, die Sie in IAM-Richtlinien zur Kontrolle des Zugriffs auf Ihre Ressourcen verwendet werden können. Die folgenden Bedingungsschlüssel stellen nur einen Teil einer IAM-Richtlinie dar. Vollständige Richtlinienbeispiele finden Sie unter Beispiele von -Zugriffspunktrichtlinien, the section called “Delegieren der Zugangskontrolle an Zugriffspunkte” und the section called “Erteilen von Berechtigungen für kontoübergreifende Zugriffspunkte”.

`s3:DataAccessPointArn`

Dieses Beispiel zeigt eine Zeichenfolge, die Sie mit einem Zugriffspunkt-ARN abgleichen können. Das folgende Beispiel entspricht allen Zugriffspunkten für AWS-Konto *123456789012* in Region *us-west-2*:

```
"Condition" : {
    "StringLike": {
        "s3:DataAccessPointArn": "arn:aws:s3:us-west-2:123456789012:accesspoint/*"
    }
}
```

`s3:DataAccessPointAccount`

Dieses Beispiel zeigt einen Zeichenfolgenoperator, mit dem Sie die Konto-ID des Besitzers eines Zugriffspunkts abgleichen können. Das folgende Beispiel entspricht allen Zugriffspunkten im Besitz des AWS-Konto s *123456789012*.

```
"Condition" : {
    "StringEquals": {
        "s3:DataAccessPointAccount": "123456789012"
    }
}
```

`s3:AccessPointNetworkOrigin`

Dieses Beispiel zeigt einen Zeichenfolgenoperator, den Sie verwenden können, um für den Netzwerkursprung entweder `Internet` oder `VPC` abzugleichen. Im folgenden Beispiel werden nur Zugriffspunkte mit einem VPC-Ursprung abgeglichen.

```
"Condition" : {
    "StringEquals": {
        "s3:AccessPointNetworkOrigin": "VPC"
    }
}
```

Weitere Informationen zur Verwendung von Bedingungsschlüsseln mit Amazon S3 finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Delegieren der Zugangskontrolle an Zugriffspunkte

Sie können die Zugriffssteuerung für einen Bucket an die Zugriffspunkte des Buckets delegieren. Die folgende Bucket-Beispielrichtlinie ermöglicht Vollzugriff auf alle Zugriffspunkte, die dem Konto des Bucket-Eigentümers gehören. Somit wird der gesamte Zugriff auf diesen Bucket durch die an seine Zugriffspunkte angehängten Richtlinien gesteuert. Wir empfehlen, Ihre Buckets auf diese Weise für alle Anwendungsfälle zu konfigurieren, die keinen direkten Zugriff auf den Bucket erfordern.

Example 6 – Bucket-Richtlinie zum Delegieren der Zugriffskontrolle an Standardzugriffspunkte

```
{
```

```
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS": "*" },
        "Action" : "*",
        "Resource" : [ "Bucket ARN", "Bucket ARN/*"],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointAccount" : "Bucket owner's account
 ID" }
        }
    }]
}
```

Erteilen von Berechtigungen für kontoübergreifende Zugriffspunkte

Wenn Sie einen Zugriffspunkt für einen Bucket erstellen möchten, der einem anderen Konto gehört, müssen Sie zuerst den Zugriffspunkt erstellen, indem Sie den Bucket-Namen und die Kontobesitzer-ID angeben. Dann muss der Bucket-Eigentümer die Bucket-Richtlinie aktualisieren, um Anfragen vom Zugriffspunkt zu autorisieren. Ein Zugriffspunkt wird insofern ähnlich erstellt wie ein DNS-CNAME, als der Zugriffspunkt keinen Zugriff auf den Bucket-Inhalt bietet. Der gesamte Bucket-Zugriff wird durch die Bucket-Richtlinie kontrolliert. Die folgende Bucket-Beispielrichtlinie erlaubt `GET`- und `LIST`-Anfragen an den Bucket von einem Zugriffspunkt aus, der einem vertrauenswürdigen AWS-Konto gehört.

Ersetzen Sie *`Bucket-ARN`* durch den ARN des Buckets.

Example 7 – Bucket-Richtlinie, die Berechtigungen an ein anderes delegiert AWS-Konto

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS": "*" },
        "Action" : ["s3:GetObject","s3:ListBucket"],
        "Resource" : [ "Bucket ARN", "Bucket ARN/*"],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointAccount" : "Access point owner's
 account ID" }
        }
    }]
```

```
}
```

Erstellen von Zugriffspunkten

Amazon S3 stellt Funktionen zum Erstellen und Verwalten von Zugriffspunkten bereit. Sie können S3-Zugriffspunkte mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDKs oder der Amazon S3-REST-API erstellen.

Standardmäßig können Sie für jedes Ihrer AWS-Konten bis zu 10 000 Zugriffspunkte pro Region erstellen. Wenn Sie mehr als 10 000 Zugriffspunkte für ein einzelnes Konto in einer Region benötigen, können Sie eine Erhöhung der Service Quotas beantragen. Weitere Informationen zu Service Quotas und zum Beantragen einer Erhöhung finden Sie unter AWS Service Quotas in der Allgemeine AWS-Referenz.

> **Note**
>
> Da Sie möglicherweise Ihren Zugriffspunktnamen veröffentlichen möchten, damit andere Benutzer den Zugriffspunkt verwenden können, sollten Sie vertrauliche Informationen im Namen des Zugriffspunkts vermeiden. Die Namen von Zugriffspunkten werden in einer öffentlich zugänglichen Datenbank veröffentlicht, die als Domain Name System (DNS) bekannt ist.

Regeln zur Benennung von Amazon S3-Zugriffspunkten

Ein Zugriffspunktname muss die folgenden Bedingungen erfüllen:

- Muss innerhalb eines einzelnen AWS-Konto und einer Region eindeutig sein
- Muss die DNS-Benennungsbeschränkungen erfüllen
- Muss mit einer Zahl oder einem Kleinbuchstaben beginnen
- Muss zwischen 3 und 50 Zeichen lang sein.
- Dürfen nicht mit einem Bindestrich (`-`) beginnen oder enden
- Dürfen keine Unterstriche (_), Großbuchstaben oder Punkte (`.`) enthalten
- Kann nicht mit dem Suffix `-s3alias` enden. Dieses Suffix ist für Zugriffspunkt-Aliasnamen reserviert. Weitere Informationen finden Sie unter Verwenden eines Alias im Bucket-Stil für Ihren S3-Bucket-Zugriffspunkt.

Informationen zum Erstellen eines Zugriffspunkts finden Sie in den folgenden Themen.

Themen

Erstellen eines Zugriffspunkts

Ein Zugriffspunkt ist genau einem Amazon S3-Bucket zugeordnet. Wenn Sie einen Bucket in Ihrem verwenden möchten AWS-Konto, müssen Sie zunächst einen Bucket erstellen. Weitere Informationen zum Erstellen von Buckets finden Sie unter Erstellen, Konfigurieren und Arbeiten mit Amazon S3-Buckets.

Sie können auch einen kontoübergreifenden Zugriffspunkt erstellen, der mit einem Bucket in einem anderen AWS-Konto verknüpft ist, sofern Sie den Bucket-Namen und die Konto-ID des Bucket-Eigentümers kennen. Wenn Sie kontoübergreifende Zugriffspunkte erstellen, erhalten Sie jedoch erst Zugriff auf Daten im Bucket, wenn Ihnen vom Bucket-Eigentümer die entsprechenden Berechtigungen erteilt wurden. Der Bucket-Eigentümer muss dem Konto des Zugriffspunktbesitzers (Ihr Konto) über die Bucket-Richtlinie Zugriff auf den Bucket gewähren. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für kontoübergreifende Zugriffspunkte.

Standardmäßig können Sie für jedes Ihrer AWS-Konten bis zu 10 000 Zugriffspunkte pro Region erstellen. Wenn Sie mehr als 10 000 Zugriffspunkte für ein einzelnes Konto in einer Region benötigen, können Sie eine Erhöhung der Service Quotas beantragen. Weitere Informationen zu Service Quotas und zum Beantragen einer Erhöhung finden Sie unter AWS Service Quotas in der Allgemeine AWS-Referenz.

Die folgenden Beispiele zeigen, wie Sie einen Zugriffspunkt mit der AWS CLI und der S3-Konsole erstellen. Informationen zum Erstellen von Zugriffspunkten mithilfe der REST-API finden Sie unter CreateAccessPoint in der API-Referenz für Amazon Simple Storage Service.

Verwenden der S3-Konsole

So erstellen Sie einen Zugriffspunkt

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie im linken Navigationsbereich Zugriffspunkte aus.
3. Wählen Sie auf der Seite Zugriffspunkte die Option Zugriffspunkt erstellen aus.
4. Geben Sie im Feld Name des Zugriffspunkts den gewünschten Namen für den Zugriffspunkt ein. Weitere Benennungen zur Benennung von Zugangspunkten finden Sie unter Regeln zur Benennung von Amazon S3-Zugriffspunkten.
5. Geben Sie unter Bucket-Name den S3-Bucket an, den Sie mit dem Zugriffspunkt verwenden möchten.

   Um in einen Bucket in Ihrem Konto zu verwenden, wählen Sie Bucket in diesem Konto auswählen aus und geben Sie den Namen des Buckets ein oder suchen Sie danach.

   Um einen Bucket in einem anderen zu verwenden AWS-Konto, wählen Sie Bucket in einem anderen Konto angeben und geben Sie die AWS-Konto ID und den Namen des Buckets ein.

   > **Note**
   >
   > Wenn Sie einen Bucket in einem anderen verwenden AWS-Konto, muss der Bucket-Eigentümer die Bucket-Richtlinie aktualisieren, um Anforderungen vom Zugriffspunkt zu autorisieren. Ein Beispiel für eine Bucket-Richtlinie finden Sie in Beispiel Erteilen von Berechtigungen für kontoübergreifende Zugriffspunkte.

6. Wählen Sie einen Ausgangspunkt des Netzwerks aus. Wenn Sie Virtual Private Cloud (VPC) auswählen, geben Sie die VPC-ID ein, die Sie mit dem Zugriffspunkt verwenden möchten.

   Weitere Informationen zu Netzwerkursprüngen für Zugangspunkte finden Sie unter Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind.
7. Wählen Sie unter Block Public Access settings for this Access Point (Block-Public-Access-Einstellungen für diesen Zugriffspunkt) die Block-Public-Access-Einstellungen, die Sie für den Zugriffspunkt anwenden möchten. Alle Einstellungen zum Blockieren des öffentlichen Zugriffs sind standardmäßig für neue Zugriffspunkte aktiviert. Wir empfehlen, alle Einstellungen aktiviert zu lassen, es sei denn, Sie wissen, dass Sie eine bestimmte Einstellung deaktivieren müssen.

   > **Note**
   >
   > Nachdem Sie einen Zugriffspunkt erstellt haben, können Sie die Einstellungen zum Blockieren des öffentlichen Zugriffs nicht mehr ändern.

Weitere Informationen zur Verwendung von Amazon S3-Block-Public-Access mit Zugriffspunkten finden Sie unter Verwalten des öffentlichen Zugriffs auf Zugriffspunkte.

8. (Optional) Geben Sie unter Access point policy (Zugriffspunktrichtlinie) – optional die Zugriffspunktrichtlinie an. Beheben Sie vor dem Speichern Ihrer Richtlinie Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge. Weitere Informationen zum Festlegen einer Zugriffspunktrichtlinie finden Sie unter Beispiele von -Zugriffspunktrichtlinien.
9. Wählen Sie Create access point (Zugriffspunkt erstellen) aus.

Verwenden der AWS CLI

Im folgenden Beispiel wird ein Zugriffspunkt mit dem Namen *example-ap* für den Bucket *DOC-EXAMPLE-BUCKET* im Konto *111122223333* erstellt. Zum Erstellen des Zugriffspunkts senden Sie eine Anfrage an Amazon S3, die Folgendes enthält:

- Den Namen des Zugriffspunkts Informationen zu Benennungsregeln finden Sie unter the section called “Regeln zur Benennung von Amazon S3-Zugriffspunkten”.
- Den Name des Buckets, dem Sie den Zugriffspunkt zuweisen möchten
- Die Konto-ID für das AWS-Konto , dem der Bucket gehört.

```
aws s3control create-access-point --name example-ap --account-id 111122223333 --
bucket DOC-EXAMPLE-BUCKET
```

Wenn Sie einen Zugriffspunkt mithilfe eines Buckets in einem anderen erstellen AWS-Konto, fügen Sie den Parameter hinzu`--bucket-account-id`. Der folgende Beispielbefehl erstellt einen Zugriffspunkt im AWS-Konto *111122223333* mithilfe des Buckets *DOC-EXAMPLE-BUCKET2*, der sich im AWS-Konto *444455556666* befindet.

```
aws s3control create-access-point --name example-ap --account-id 111122223333 --
bucket DOC-EXAMPLE-BUCKET --bucket-account-id 444455556666
```

Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind

Wenn Sie einen Zugriffspunkt erstellen, können Sie ihn über das Internet zugänglich machen, oder Sie können angeben, dass alle über diesen Zugriffspunkt eingehenden Anforderungen aus

einer bestimmten Virtual Private Cloud (VPC) stammen müssen. Ein Zugangspunkt, der über das Internet zugänglich ist, soll einen Netzwerkursprung von `Internet` haben. Er kann von überall im Internet verwendet werden, vorbehaltlich anderer Zugriffsbeschränkungen für den Zugriffspunkt, den zugrunde liegenden Bucket und die zugehörige Ressourcen, z. B. die angeforderten Objekte. Ein Zugriffspunkt, auf den nur von einer angegebenen VPC aus zugegriffen werden kann, hat den Netzwerkursprung `VPC`. Amazon S3 weist jede Anforderung an den Zugriffspunkt zurück, die nicht von dieser VPC stammt.

> **Important**
>
> Sie können den Netzwerkursprung eines Zugriffspunkts nur angeben, wenn Sie den Zugriffspunkt erstellen. Nachdem Sie den Zugriffspunkt erstellt haben, können Sie seinen Netzwerkursprung nicht mehr ändern.

Um einen Zugriffspunkt auf reinen VPC-Zugriff zu beschränken, fügen Sie den Parameter `VpcConfiguration` in die Anforderung zum Erstellen des Zugriffspunkts ein. Im Parameter `VpcConfiguration` geben Sie die VPC-ID an, die in der Lage sein soll, den Zugriffspunkt zu verwenden. Wenn eine Anfrage über den Zugriffspunkt erfolgt, muss die Anfrage von der VPC stammen, sonst lehnt Amazon S3 sie ab.

Sie können den Netzwerkursprung eines Zugriffspunkts mithilfe der AWS CLI, AWS SDKs oder REST-APIs abrufen. Wenn für einen Zugriffspunkt eine VPC-Konfiguration angegeben ist, lautet der Netzwerkursprung `VPC`. Andernfalls ist der Netzwerkursprung des Zugriffspunkts `Internet`.

Example

Beispiel: Erstellen eines Zugriffspunkts, der auf VPC-Zugriff beschränkt ist

Im folgenden Beispiel wird ein Zugriffspunkt mit dem Namen `example-vpc-ap` für Bucket `example-bucket` in Konto `123456789012` erstellt, der den Zugriff nur von der VPC `vpc-1a2b3c` aus zulässt. Das Beispiel überprüft dann, ob der neue Zugriffspunkt den Netzwerkursprung `VPC` hat.

AWS CLI

```
aws s3control create-access-point --name example-vpc-ap --account-id 123456789012 --
bucket example-bucket --vpc-configuration VpcId=vpc-1a2b3c
```

```
aws s3control get-access-point --name example-vpc-ap --account-id 123456789012
```

```
{
    "Name": "example-vpc-ap",
    "Bucket": "example-bucket",
    "NetworkOrigin": "VPC",
    "VpcConfiguration": {
        "VpcId": "vpc-1a2b3c"
    },
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": true,
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2019-11-27T00:00:00Z"
}
```

Um einen Zugriffspunkt mit einer VPC zu verwenden, müssen Sie die Zugriffsrichtlinie für Ihren VPC-Endpunkt ändern. VPC-Endpunkte ermöglichen den Datenfluss von Ihrer VPC zu Amazon S3. Sie verfügen über Zugriffssteuerungsrichtlinien, die kontrollieren, wie Ressourcen innerhalb der VPC mit Amazon S3 interagieren dürfen. Anforderungen von Ihrer VPC an Amazon S3 werden nur dann über einen Zugriffspunkt erfolgreich ausgeführt, wenn die VPC-Endpunktrichtlinie sowohl Zugriff auf den Zugriffspunkt als auch auf den zugrunde liegenden Bucket gewährt.

> **Note**
>
> Damit Ressourcen nur innerhalb einer VPC zugänglich sind, erstellen Sie unbedingt eine privat gehostete Zone für Ihren VPC-Endpunkt. Wenn Sie eine privat gehostete Zone verwenden möchten, ändern Sie Ihre VPC-Einstellungen so, dass die VPC-Netzwerkattribute `enableDnsHostnames` und `enableDnsSupport` auf `true` festgelegt sind.

In der folgenden Beispielrichtlinienanweisung wird ein VPC-Endpunkt so konfiguriert, dass Aufrufe von `GetObject` für einen Bucket mit dem Namen `awsexamplebucket1` und einen Zugriffspunkt mit dem Namen `example-vpc-ap` zulässig sind.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Principal": "*",
```

```
        "Action": [
            "s3:GetObject"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::awsexamplebucket1/*",
            "arn:aws:s3:us-west-2:123456789012:accesspoint/example-vpc-ap/object/*"
        ]
    }]
}
```

> **Note**
>
> Die `"Resource"`-Deklaration in diesem Beispiel verwendet einen Amazon-Ressourcennamen (ARN) zur Angabe des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-ARNs finden Sie unter Verwenden von Zugriffspunkten.

Weitere Informationen zu VPC-Endpunktrichtlinien finden Sie unter Verwenden von Endpoint-Richtlinien für Amazon S3 im VPC-Benutzerhandbuch.

Verwalten des öffentlichen Zugriffs auf Zugriffspunkte

Amazon S3-Zugriffspunkte unterstützen unabhängige Block Public Access-Einstellungen für jeden Zugriffspunkt. Wenn Sie einen Zugriffspunkt erstellen, können Sie die Block Public Access-Einstellungen für diesen Zugriffspunkt festlegen. Für jede Anforderung, die über einen Zugriffspunkt eingeht, wertet Amazon S3 die Block Public Access-Einstellungen für diesen Zugriffspunkt, den zugrunde liegenden Bucket und das Konto des Bucket-Eigentümers aus. Wenn eine dieser Einstellungen darauf hinweist, dass die Anforderung gesperrt werden soll, lehnt Amazon S3 die Anforderung ab.

Weitere Hinweise zur Funktion S3 Block Public Access finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

> **Important**
>
> - Alle Block Public Access-Einstellungen sind standardmäßig für Zugriffspunkte aktiviert. Sie müssen alle Einstellungen, die Sie nicht auf einen Zugriffspunkt anwenden möchten, explizit deaktivieren.

- Amazon S3 unterstützt derzeit nicht das Ändern der Public Block Access-Einstellungen eines Zugriffspunkts, nachdem der Zugriffspunkt erstellt wurde.

Example

Beispiel: Erstellen eines Zugriffspunkts mit benutzerdefinierten Block-Public-Access-Einstellungen

In diesem Beispiel wird ein Zugriffspunkt mit dem Namen `example-ap` für Bucket `example-bucket` in Konto `123456789012` mit nicht standardmäßigen Block Public Access-Einstellungen erstellt. Das Beispiel ruft dann die Konfiguration des neuen Zugriffspunkts ab, um seine Block Public Access-Einstellungen zu überprüfen.

AWS CLI

```
aws s3control create-access-point --name example-ap --account-id
 123456789012 --bucket example-bucket --public-access-block-configuration
 BlockPublicAcls=false,IgnorePublicAcls=false,BlockPublicPolicy=true,RestrictPublicBuckets=t
```

```
aws s3control get-access-point --name example-ap --account-id 123456789012

{
    "Name": "example-ap",
    "Bucket": "example-bucket",
    "NetworkOrigin": "Internet",
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": false,
        "IgnorePublicAcls": false,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2019-11-27T00:00:00Z"
}
```

Verwenden von Zugriffspunkten

Sie können auf die Objekte in einem Amazon S3-Bucket mit einem Zugriffspunkt über die AWS Management Console AWS CLI, , AWS SDKs oder die S3-REST-APIs zugreifen.

Zugriffspunkte haben Amazon-Ressourcennamen (ARN). Zugriffspunkt-ARNs ähneln Bucket-ARNs, werden jedoch explizit eingegeben und kodieren die Region des Zugriffspunkts und die AWS-Konto -ID des Eigentümers des Zugriffspunkts. Weitere Informationen zur Verwendung von ARNs finden Sie unter Amazon-Ressourcennamen (ARN) im Allgemeine AWS-Referenz.

Zugriffspunkt-ARNs verwenden das Format `arn:aws:s3:region:account-id:accesspoint/resource`. z. B.:

- arn:aws:s3:us-west- 2:123456789012:accesspoint/test repräsentiert den Zugriffspunkt `test` im Besitz von Konto `123456789012` in Region `us-west-2`.
- arn:aws:s3:us-west- 2:123456789012:accesspoint/* repräsentiert alle Zugriffspunkte unter Konto `123456789012` in Region `us-west-2`.

ARNs für Objekte, auf die über einen Zugriffspunkt zugegriffen wird, haben das Format `arn:aws:s3:region:account-id:accesspoint/access-point-name/object/resource`. z. B.:

- arn:aws:s3:us-west- 2:123456789012:accesspoint/test/object/unit-01 repräsentiert das Objekt `unit-01`, auf das über den Zugriffspunkt `test` im Besitz von Konto `123456789012` in Region `us-west-2` zugegriffen wird.
- arn:aws:s3:us-west- 2:123456789012:accesspoint/test/object/* repräsentiert alle Objekte für Zugriffspunkt `test` in Konto `123456789012` in Region `us-west-2`.
- arn:aws:s3:us-west- 2:123456789012:accesspoint/test/object/unit-01/finance/* repräsentiert alle Objekte unter Präfix `unit-01/finance/` für Zugriffspunkt `test` in Konto `123456789012` und Region `us-west-2`.

Themen

- Überwachen und Protokollieren von Zugriffspunkten
- Nutzen von Amazon S3-Zugriffspunkten mit der Amazon S3-Konsole
- Verwenden eines Alias im Bucket-Stil für Ihren S3-Bucket-Zugriffspunkt
- Verwenden von Zugriffspunkten mit kompatiblen Amazon S3-Vorgänge

Wenn Sie über eine Virtual Private Cloud (VPC) verfügen, finden Sie weitere Informationen unter Verwalten des Amazon-S3-Zugriffs mit VPC-Endpunkten und S3-Zugriffspunkten.

Überwachen und Protokollieren von Zugriffspunkten

Amazon S3 protokolliert Anforderungen, die über Zugriffspunkte ausgeführt werden, und Anforderungen an die APIs, die Zugriffspunkte verwalten, z. B. `CreateAccessPoint` und `GetAccessPointPolicy`. Um Nutzungsmuster zu überwachen und zu verwalten, können Sie auch Amazon- CloudWatch Logs-Anforderungsmetriken für Zugriffspunkte konfigurieren.

Themen

CloudWatch -Anforderungsmetriken

Um die Leistung von Anwendungen zu verstehen und zu verbessern, die Zugriffspunkte verwenden, können Sie CloudWatch für Amazon S3-Anforderungsmetriken verwenden. Anforderungsmetriken helfen Ihnen beim Überwachen von Amazon S3-Anforderungen, um Probleme bei der Ausführung schnell zu identifizieren und zu beheben.

Standardmäßig stehen diese Anforderungsmetriken auf Ebene der Buckets zur Verfügung. Sie können jedoch einen Filter für Anforderungsmetriken mit einem gemeinsamen Präfix, Objekt-Tags oder einem Zugriffspunkt definieren. Wenn Sie einen Zugriffspunktfilter erstellen, enthält die Konfiguration der Anforderungsmetriken Anforderungen an den von Ihnen angegebenen Zugriffspunkt. Sie können Metriken erhalten, Alarme festlegen und auf Dashboards zugreifen, um Echtzeit-Vorgänge anzuzeigen, die über diesen Zugriffspunkt ausgeführt werden.

Sie wählen Anforderungsmetriken aus, indem Sie diese in der Konsole konfigurieren oder die Amazon S3-API verwenden. Die Anforderungsmetriken sind in 1-Minuten-Intervallen nach einer gewissen Latenz für die Verarbeitung verfügbar. Anforderungsmetriken werden zum gleichen Tarif wie CloudWatch benutzerdefinierte Metriken abgerechnet. Weitere Informationen finden Sie unter Amazon- CloudWatch Preise.

Informationen zum Erstellen einer Anforderungsmetrikkonfiguration, die nach Zugriffspunkt filtert, finden Sie unter Erstellen einer Metrik-Konfiguration, die nach dem Präfix, Objekt-Tag oder Zugriffspunkt filtert.

Anforderungsprotokolle

Sie können Anforderungen über Zugriffspunkte und Anforderungen an die APIs, die Zugriffspunkte verwalten, protokollieren, z. B. `CreateAccessPoint` und `GetAccessPointPolicy`,, indem Sie die Serverzugriffsprotokollierung und AWS CloudTrail nutzen.

CloudTrail -Protokolleinträge für Anforderungen, die über Zugriffspunkte gestellt werden, enthalten den Zugriffspunkt-ARN im -`resources`Abschnitt des Protokolls.

Angenommen, folgende Konfiguration liegt vor:

- Ein Bucket mit dem Namen `DOC-EXAMPLE-BUCKET1` in Region `us-west-2`, das ein Objekt namens `my-image.jpg` enthält
- Ein Zugriffspunkt mit dem Namen `my-bucket-ap`, der mit `DOC-EXAMPLE-BUCKET1` verknüpft ist
- Eine AWS-Konto ID von `123456789012`

Das folgende Beispiel zeigt den -`resources`Abschnitt eines CloudTrail Protokolleintrags für die vorherige Konfiguration:

```
"resources": [
        {"type": "AWS::S3::Object",
            "ARN": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/my-image.jpg"
        },
        {"accountId": "123456789012",
            "type": "AWS::S3::Bucket",
            "ARN": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
        },
        {"accountId": "123456789012",
            "type": "AWS::S3::AccessPoint",
            "ARN": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-bucket-ap"
        }
    ]
```

Weitere Informationen zu S3 Server-Zugriffsprotokollen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung. Weitere Informationen zu AWS CloudTrail finden Sie unter Was ist AWS CloudTrail? im AWS CloudTrail -Benutzerhandbuch.

Nutzen von Amazon S3-Zugriffspunkten mit der Amazon S3-Konsole

In diesem Abschnitt wird erläutert, wie Sie Ihre Amazon S3 Access Points mithilfe der AWS Management Console verwalten und verwenden. Bevor Sie beginnen, navigieren Sie zur Detailseite für den Zugriffspunkt, den Sie verwalten oder verwenden möchten, wie im folgenden Verfahren beschrieben.

Themen

Auflisten von Zugriffspunkten für Ihr Konto

So listen Sie alle Zugriffspunkte auf, die in Ihrem erstellt wurden AWS-Konto

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie im Navigationsbereich links in der Konsole auf Access points (Zugriffspunkte).
3. Wählen Sie auf der Seite Zugriffspunkte unter Zugriffspunkte die aus, AWS-Region die die Zugriffspunkte enthält, die Sie auflisten möchten.
4. (Optional) Suchen Sie nach Zugriffspunkten nach Name, indem Sie einen Namen in das Textfeld neben dem Dropdown-Menü „Region“ eingeben.
5. Wählen Sie den Namen des Zugriffspunkts, den Sie verwalten oder verwenden möchten.

Auflisten von Zugriffspunkten für einen Bucket

So listen Sie alle Zugriffspunkte in Ihnen AWS-Konto für einen einzelnen Bucket auf

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Klicken Sie im Navigationsbereich auf der linken Seite der Konsole auf Buckets.
3. Wählen Sie auf der Seite Buckets den Namen des Buckets aus, dessen Zugriffspunkte Sie auflisten möchten.
4. Wählen Sie auf der Bucket-Detailseite den Tab Access points (Zugriffspunkte).
5. Wählen Sie den Namen des Zugriffspunkts, den Sie verwalten oder verwenden möchten.

Anzeigen der Konfigurationsdetails für einen Zugriffspunkt

1. Navigieren Sie zur Detailseite des Zugriffspunkts für den Zugriffspunkt, dessen Details Sie anzeigen möchten, wie unter Auflisten von Zugriffspunkten für Ihr Konto beschrieben.
2. Zeigen Sie unter Access point overview (Zugriffspunkt-Übersicht) die Konfigurationsdetails und Eigenschaften für den ausgewählten Zugriffspunkt an.

Verwenden eines Zugriffspunkts

1. Navigieren Sie zur Detailseite des Zugriffspunkts für den gewünschten Zugriffspunkt, wie unter Auflisten von Zugriffspunkten für Ihr Konto beschrieben.
2. Wählen Sie im Tab Objects (Objekte) den Namen eines Objekts oder von Objekten aus, auf das/die Sie über den Zugriffspunkt zugreifen möchten. Die Konsole zeigt über dem Namen Ihres Buckets eine Beschriftung mit dem aktuell von Ihnen verwendeten Zugriffspunkt an. Während Sie den Zugriffspunkt verwenden, können Sie nur die Objekt-Vorgänge ausführen, die durch die Zugriffspunktberechtigungen gewährt werden.

> **Note**
>
> - In der Konsolenansicht werden immer alle Objekte im Bucket angezeigt. Die Verwendung eines Zugriffspunkts, wie in dieser Prozedur beschrieben, beschränkt die Vorgänge, die Sie für diese Objekte ausführen können, jedoch nicht, ob Sie sehen können, dass sie im Bucket vorhanden sind.
> - Die S3 Management Console unterstützt nicht die Verwendung von VPC-Zugriffspunkten (Virtual Private Cloud) für den Zugriff auf Bucket-Ressourcen. Um von einem VPC-Zugriffspunkt aus auf Bucket-Ressourcen zuzugreifen, verwenden Sie die AWS CLI, AWS SDKs oder Amazon S3-REST-APIs .

Anzeigen der Block Public Access-Einstellungen für einen Zugriffspunkt

1. Navigieren Sie zur Detailseite des Zugriffspunkts für den Zugriffspunkt, dessen Einstellungen Sie anzeigen möchten, wie unter Auflisten von Zugriffspunkten für Ihr Konto beschrieben.
2. Wählen Sie Permissions (Berechtigungen).
3. Überprüfen Sie unter Access point policy (Zugriffspunktrichtlinie) die Block-Public-Access-Einstellungen für den Zugriffspunkt.

> **Note**
>
> Sie können die Block Public Access-Einstellungen für einen Zugriffspunkt nicht ändern, nachdem der Zugriffspunkt erstellt wurde.

Bearbeiten einer Zugriffspunktrichtlinie

1. Navigieren Sie zur Detailseite des Zugriffspunkts für den Zugriffspunkt, dessen Richtlinie Sie bearbeiten möchten, wie unter Auflisten von Zugriffspunkten für Ihr Konto beschrieben.
2. Wählen Sie Permissions (Berechtigungen).
3. Wählen Sie unter Access point policy (Zugriffspunktrichtlinie) Edit (Bearbeiten) aus.
4. Geben Sie die Richtlinie für den Zugriffspunkt in das Textfeld ein. Die Konsole zeigt automatisch den Amazon-Ressourcennamen (ARN) für den Zugriffspunkt an, den Sie in der Richtlinie verwenden können.

Erstellen eines Zugriffspunkts

1. Navigieren Sie zur Liste der Zugriffspunkte für Ihr Konto oder für einen bestimmten Bucket, wie unter Auflisten von Zugriffspunkten für Ihr Konto beschrieben.
2. Aktivieren Sie die Optionsschaltfläche neben dem Namen des Zugriffspunkts, den Sie löschen möchten.
3. Wählen Sie Delete (Löschen).
4. Bestätigen Sie, dass Sie Ihren Zugriffspunkt löschen möchten, indem Sie den Namen in das angezeigte Textfeld eingeben und Delete (Löschen) wählen.

Verwenden eines Alias im Bucket-Stil für Ihren S3-Bucket-Zugriffspunkt

Wenn Sie einen Zugriffspunkt erstellen, generiert Amazon S3 automatisch einen Alias, den Sie anstelle eines Amazon S3-Bucket-Namens für den Datenzugriff verwenden können. Sie können diesen Zugriffspunkt-Alias anstelle eines Amazon-Ressourcennamens (ARN) für Zugriffspunkt-Operationen auf Datenebene verwenden. Eine Liste dieser Vorgänge finden Sie unter Kompatibilität von Zugriffspunkten mit - AWS Services.

Das folgende Beispiel zeigt einen ARN- und Zugriffspunkt-Alias für einen Zugriffspunkt namens *my-access-point*.

- ARN – `arn:aws:s3:`*region*`:`*account-id*`:accesspoint/`*my-access-point*
- Zugriffspunkt-Alias – *my-access-point*`-`*hrzrlukc5m36ft7okagglf3gmwluquse1b*`-s3alias`

Weitere Informationen zur Verwendung von ARNs finden Sie unter Amazon-Ressourcennamen (ARN) im Allgemeine AWS-Referenz.

Aliasname für Zugriffspunkte

Ein Zugriffspunkt-Aliasname wird innerhalb desselben Namespace wie ein Amazon S3-Bucket erstellt. Dieser Aliasname wird automatisch generiert und kann nicht geändert werden. Ein Zugriffspunkt-Aliasname erfüllt alle Anforderungen eines gültigen Amazon S3-Bucket-Namens und besteht aus den folgenden Teilen:

access point prefix`-`*metadata*`-s3alias`

> **Note**
>
> Das Suffix `-s3alias` ist für Zugriffspunkt-Aliasnamen reserviert und kann nicht für Bucket- oder Zugriffspunkt-Namen verwendet werden. Weitere Informationen zu Amazon-S3-Bucket-Benennungsregeln finden Sie unter Regeln für die Benennung von Buckets.

Zugriffspunkt-Alias-Anwendungsfälle und -beschränkungen

Bei der Übernahme von Zugriffspunkten können Sie Zugriffspunkt-Aliasnamen verwenden, ohne dass umfangreiche Codeänderungen erforderlich sind.

Wenn Sie einen Zugriffspunkt erstellen, generiert Amazon S3 automatisch einen Zugriffspunkt-Aliasnamen, wie im folgenden Beispiel gezeigt. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control create-access-point --bucket DOC-EXAMPLE-BUCKET1 --name my-access-point
 --account-id 111122223333
{
    "AccessPointArn":
    "arn:aws:s3:region:111122223333:accesspoint/my-access-point",
    "Alias": "my-access-point-aqfqprnstn7aefdfbarligizwgyfouse1a-s3alias"
}
```

Sie können diesen Zugriffspunkt-Aliasnamen anstelle eines Amazon S3-Bucket-Namen für jeden Datenebenen-Vorgang verwenden. Eine Liste dieser Vorgänge finden Sie unter Kompatibilität von Zugriffspunkten mit - AWS Services.

Im folgenden AWS CLI Beispiel für den `get-object` Befehl wird der Zugriffspunkt-Alias des Buckets verwendet, um Informationen über das angegebene Objekt zurückzugeben. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3api get-object --bucket my-access-point-aqfqprnstn7aefdfbarligizwgyfouse1a-
s3alias --key dir/my_data.rtf my_data.rtf

{
    "AcceptRanges": "bytes",
    "LastModified": "2020-01-08T22:16:28+00:00",
    "ContentLength": 910,
    "ETag": "\"00751974dc146b76404bb7290f8f51bb\"",
    "VersionId": "null",
    "ContentType": "text/rtf",
    "Metadata": {}
}
```

Einschränkungen

- Aliase können nicht von Kunden konfiguriert werden.
- Aliasse können auf einem Zugriffspunkt nicht gelöscht, geändert oder deaktiviert werden.
- Sie können diesen Zugriffspunkt-Aliasnamen anstelle eines Amazon-S3-Bucket-Namen für manche Datenebenen-Operationen verwenden. Eine Liste dieser Vorgänge finden Sie unter Zugriffspunkt-Kompatibilität mit S3-Vorgänge.

- Sie können einen Aliasnamen für Amazon S3-Kontrollebenenvorgänge nicht verwenden. Eine Liste der Amazon S3-Kontrollebenenvorgänge finden Sie unter Amazon S3-Kontrolle in der API-Referenz für Amazon Simple Storage Service.
- Aliase können nicht in AWS Identity and Access Management (IAM)-Richtlinien verwendet werden.
- Aliase können nicht als Protokollierungsziel für S3-Server-Zugriffsprotokolle verwendet werden.
- Aliase können nicht als Protokollierungsziel für AWS CloudTrail Protokolle verwendet werden.
- Amazon unterstützt SageMaker GroundTruth keine Zugriffspunkt-Aliase.

Verwenden von Zugriffspunkten mit kompatiblen Amazon S3-Vorgänge

Die folgenden Beispiele veranschaulichen, wie Zugriffspunkte mit kompatiblen Vorgängen in Amazon S3 verwendet werden.

Themen

Kompatibilität von Zugriffspunkten mit - AWS Services

Mit Aliasnamen von Amazon-S3-Zugriffspunkten können Anwendungen, die einen S3-Bucketnamen benötigen, problemlos einen Zugriffspunkt verwenden. Sie können S3-Zugriffspunkt-Aliasnamen überall verwenden, wo Sie S3-Bucketnamen verwenden, um auf Daten in S3 zuzugreifen. Weitere Informationen finden Sie unter Zugriffspunkt-Alias-Anwendungsfälle und -beschränkungen.

Zugriffspunkt-Kompatibilität mit S3-Vorgänge

Sie können Zugriffspunkte verwenden, um unter Verwendung der folgenden Teilmenge von Amazon S3-APIs auf einen Bucket zuzugreifen. Alle unten aufgeführten Vorgänge können entweder Zugriffspunkt-ARNs oder Zugriffspunkt-Aliase akzeptieren:

S3-Operationen

- AbortMultipartUpload
- CompleteMultipartUpload
- CopyObject (nur Kopien in derselben Region)
- CreateMultipartUpload
- DeleteObject
- DeleteObjectTagging
- GetBucketAcl
- GetBucketCors
- GetBucketLocation
- GetBucketNotificationConfiguration
- GetBucketPolicy
- GetObject
- GetObjectAcl
- GetObjectLegalHold
- GetObjectRetention
- GetObjectTagging
- HeadBucket
- HeadObject
- ListMultipartUploads
- ListObjects
- ListObjectsV2
- ListParts
- Presign
- PutObject
- PutObjectLegalHold
- PutObjectRetention
- PutObjectAcl

- PutObjectTagging
- RestoreObject
- UploadPart
- UploadPartCopy (nur Kopien in derselben Region)

Anfordern eines Objekts über einen Zugriffspunkt

Das folgende Beispiel veranschaulicht, wie Sie das Objekt `my-image.jpg` über den Zugriffspunkt `prod` anfordern, der der Konto-ID `123456789012` in Region `us-west-2` gehört und die heruntergeladene Datei unter `download.jpg` speichert.

AWS CLI

```
aws s3api get-object --key my-image.jpg --bucket arn:aws:s3:us-
west-2:123456789012:accesspoint/prod download.jpg
```

Hochladen eines Objekts über einen Zugriffspunkt-Alias

Im folgenden Beispiel wird das Objekt `my-image.jpg` über den Zugriffspunkt-Alias `my-access-point-hrzrlukc5m36ft7okagglf3gmwluquse1b-s3alias` hochgeladen, der der Konto-ID `123456789012` in Region `us-west-2` gehört.

AWS CLI

```
aws s3api put-object --bucket my-access-point-hrzrlukc5m36ft7okagglf3gmwluquse1b-
s3alias --key my-image.jpg --body my-image.jpg
```

Löschen eines Objekts über einen Zugriffspunkt

Im folgenden Beispiel wird veranschaulicht, wie das Objekt `my-image.jpg` über den Zugriffspunkt `prod` gelöscht wird, das Konto-ID `123456789012` in Region `us-west-2` gehört.

AWS CLI

```
aws s3api delete-object --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod
 --key my-image.jpg
```

Auflisten von Objekten über einen Zugriffspunkt-Alias

Im folgenden Beispiel werden Objekte über den Zugriffspunkt-Alias `my-access-point-hrzrlukc5m36ft7okagglf3gmwluquse1b-s3alias` aufgelistet, der der Konto-ID `123456789012` in Region `us-west-2` gehört.

AWS CLI

```
aws s3api list-objects-v2 --bucket my-access-point-
hrzrlukc5m36ft7okagglf3gmwluquse1b-s3alias
```

Hinzufügen eines Tag-Satzes zu einem Objekt über einen Zugriffspunkt

Das folgende Beispiel veranschaulicht, wie Sie dem vorhandenen Objekt `my-image.jpg` über den Zugriffspunkt `prod`, der von Konto-ID `123456789012` in Region `us-west-2` gehört, einen Tag-Satz hinzufügen.

AWS CLI

```
aws s3api put-object-tagging --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/
prod --key my-image.jpg --tagging TagSet=[{Key="finance",Value="true"}]
```

Gewähren von Zugriffsberechtigungen über einen Zugriffspunkt mithilfe einer ACL

Im folgenden Beispiel wird veranschaulicht, wie eine ACL auf das vorhandene Objekt `my-image.jpg` über den Zugriffspunkt `prod`, der Konto-ID `123456789012` in Region `us-west-2` gehört, angewandt wird.

AWS CLI

```
aws s3api put-object-acl --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod
 --key my-image.jpg --acl private
```

Einschränkungen von Access Points

Es gelten die folgenden Einschränkungen und Beschränkungen für Amazon S3-Zugriffspunkte:

- Jeder Zugriffspunkt ist genau einem Bucket zugeordnet, den Sie beim Erstellen des Zugriffspunkts angeben müssen. Nachdem Sie einen Zugriffspunkt erstellt haben, können Sie ihn keinem anderen Bucket zuordnen. Sie können jedoch einen Zugriffspunkt löschen und dann einen anderen mit demselben Namen erstellen. Diesen neuen Zugriffspunkt können Sie dann einem anderen Bucket zuordnen.
- Namen von Zugriffspunkten müssen bestimmte Bedingungen erfüllen. Weitere Informationen zur Benennung von Zugangspunkten finden Sie unter Regeln zur Benennung von Amazon S3-Zugriffspunkten.
- Nachdem Sie einen Zugriffspunkt erstellt haben, können Sie die VPC-Konfiguration (Virtual Private Cloud) nicht mehr ändern.
- Zugriffspunkt-Richtlinien sind auf eine Größe von 20 KB beschränkt.
- Sie können maximal 10.000 Zugriffspunkte AWS-Konto pro Region erstellen. Wenn Sie mehr als 10 000 Zugriffspunkte für ein einzelnes Konto in einer Region benötigen, können Sie eine Erhöhung der Service Quotas beantragen. Weitere Informationen zu Service Quotas und zum Beantragen einer Erhöhung finden Sie unter AWS Service Quotas in der Allgemeine AWS-Referenz.
- In , in AWS-Regionen denen Sie über mehr als 1 000 Zugriffspunkte verfügen, können Sie in der Amazon S3-Konsole nicht nach einem Zugriffspunkt anhand des Namens suchen.
- Sie können einen Zugriffspunkt nicht als Ziel für die Replikation in S3 verwenden. Weitere Informationen zur Replikation finden Sie unter Replizieren von Objekten.
- Sie können Zugriffspunkte nur mithilfe von virtual-host-style URLs adressieren. Weitere Informationen zur virtual-host-style Adressierung finden Sie unter Zugreifen auf einen Amazon-S3-Bucket und Auflisten des Buckets.
- API-Operationen, die die Zugriffspunktfunktionalität steuern (z. B. `PutAccessPoint` und `GetAccessPointPolicy`), unterstützen keine kontoübergreifende Aufrufe.
- Sie müssen AWS Signature Version 4 verwenden, wenn Sie Anforderungen an einen Zugriffspunkt über die REST-APIs stellen. Weitere Informationen zum Authentifizieren von Anforderungen finden Sie unter Authentifizieren von Anforderungen (AWS Signature Version 4) in der Amazon Simple Storage Service API-Referenz.
- Zugriffspunkte unterstützen nur den Zugriff über HTTPS.
- Zugriffspunkte unterstützen keinen anonymen Zugriff.
- Mit den kontoübergreifenden Zugriffspunkten erhalten Sie jedoch erst dann Zugriff auf Daten, wenn Ihnen vom Bucket-Eigentümer die entsprechenden Berechtigungen erteilt wurden. Der Bucket-Eigentümer behält immer die ultimative Kontrolle über den Datenzugriff und muss die

Bucket-Richtlinie aktualisieren, um Anforderungen vom kontoübergreifenden Zugriffspunkt zu autorisieren. Eine Bucket-Beispielrichtlinie finden Sie unter Konfigurieren von IAM-Richtlinien für die Verwendung von Zugriffspunkten.

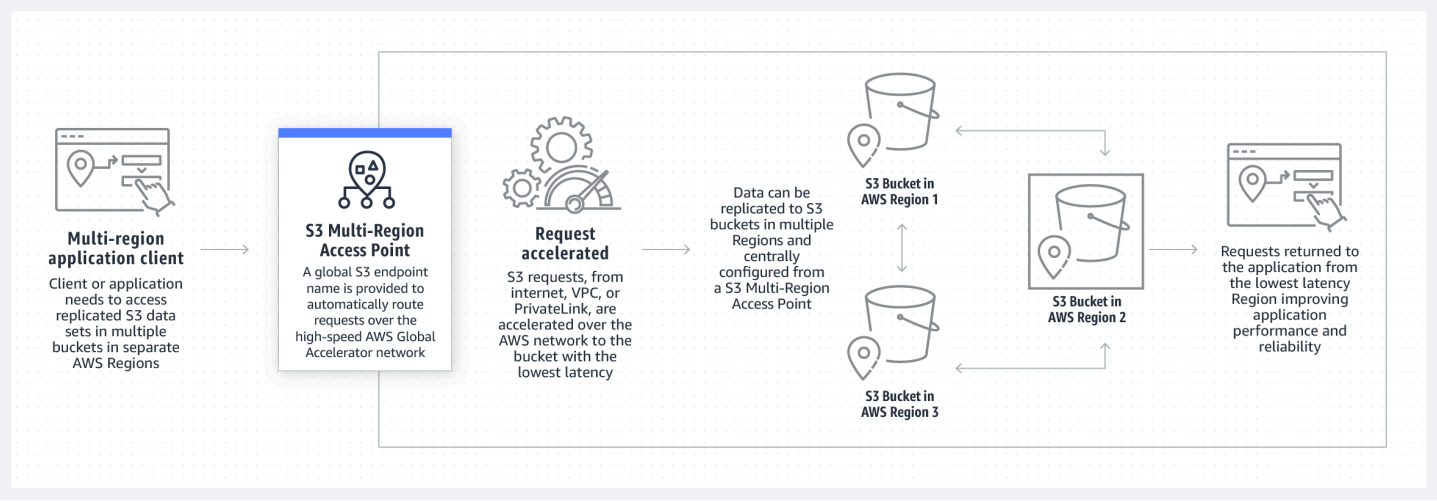
- Wenn Sie in der Amazon-S3-Konsole einen kontoübergreifenden Zugriffspunkt anzeigen, wird in der Spalte Zugriff Unbekannt angezeigt. Die Amazon-S3-Konsole kann nicht feststellen, ob öffentlicher Zugriff für den zugehörigen Bucket und die zugehörigen Objekte gewährt wurde. Sofern Sie keine öffentliche Konfiguration für einen spezifischen Anwendungsfall benötigen, empfehlen wir, dass Sie und der Bucket-Eigentümer den gesamten öffentlichen Zugriff auf den Zugriffspunkt und den Bucket blockieren. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Multi-Regions-Zugriffspunkte in Amazon S3

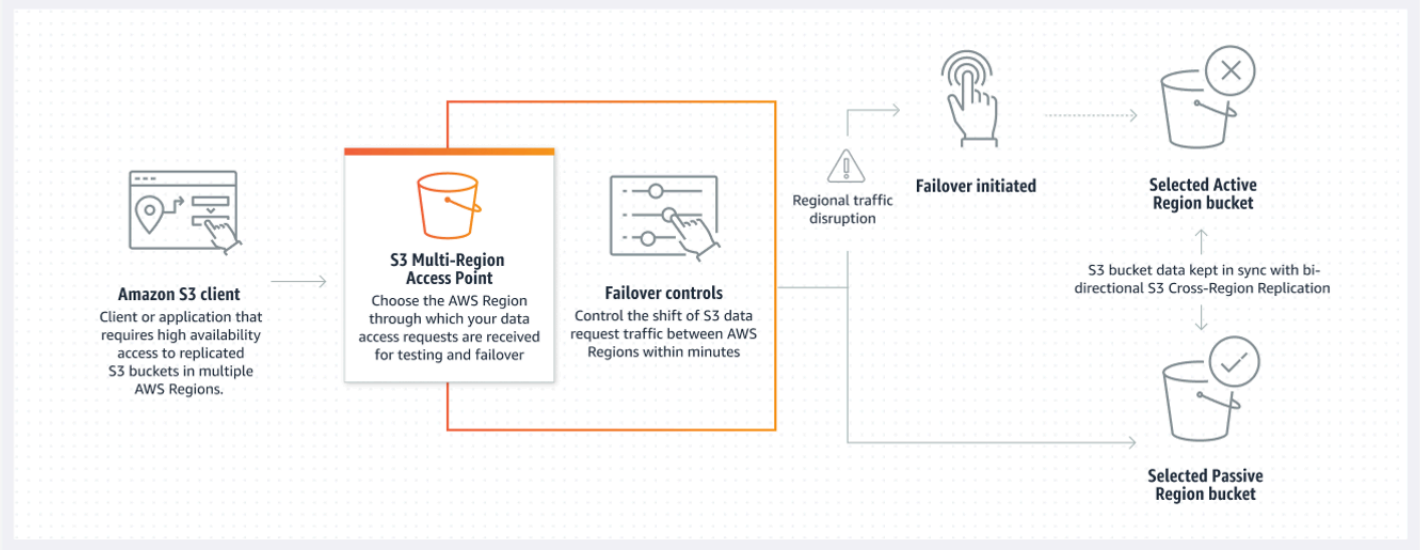
Amazon S3 Multi-Region Access Points stellen einen globalen Endpunkt bereit, mit dem Anwendungen Anforderungen von S3-Buckets ausführen können, die sich in mehreren AWS-Regionen befinden. Sie können Multi-Regions Access Points verwenden, um Multi-Regions-Anwendungen mit derselben Architektur zu erstellen, die in einer einzelnen Region verwendet wird, und diese Anwendungen dann überall auf der Welt ausführen. Anstatt Anforderungen über das überlastete öffentliche Internet zu senden, bieten Multi-Regions-Zugriffspunkte integrierte Netzwerkausfallsicherheit mit Beschleunigung internetbasierter Anforderungen an Amazon S3. Anwendungsanforderungen an einen globalen Endpunkt von Multi-Region Access Points verwenden AWS Global Accelerator für die automatische Weiterleitung über das globale AWS-Netzwerk an den nächstgelegenen S3-Bucket mit einem aktiven Routing-Status.

Wenn Sie einen Multi-Region Access Point erstellen, geben Sie eine Gruppe von AWS-Regionen an, in denen Daten gespeichert werden sollen, die über diesen Multi-Region Access Point bereitgestellt werden sollen. Sie können S3-regionsübergreifende Replikation (CRR) verwenden, um Daten zwischen Buckets in diesen Regionen zu synchronisieren. Anschließend können Sie Daten über den globalen Multi-Regions-Zugriffspunkt-Endpunkt anfordern oder schreiben. Amazon S3 übermittelt Anforderungen automatisch an den replizierten Datensatz aus der nächstgelegenen verfügbaren Region. Multi-Region Access Points sind auch mit Anwendungen kompatibel, die in Amazon Virtual Private Clouds (VPCs) ausgeführt werden, einschließlich Anwendungen, die AWS PrivateLink für Amazon S3 verwenden.

Die folgende Abbildung ist eine grafische Darstellung eines Amazon S3 Multi-Region Access Points in einer Aktiv-Aktiv-Konfiguration. Die Grafik zeigt, wie Amazon-S3-Anforderungen automatisch an Buckets in der nächstgelegenen aktiven AWS-Region weitergeleitet werden.



Die folgende Abbildung ist eine grafische Darstellung eines Amazon S3 Multi-Region Access Points in einer Aktiv-Passiv-Konfiguration. Die Grafik zeigt, wie Sie den Amazon-S3-Datenzugriffsverkehr so steuern können, dass ein Failover zwischen aktiven und passiven AWS-Regionen erfolgt.



Weitere Informationen zur Verwendung von Multi-Region Access Points finden Sie unter Tutorial: Getting started with Amazon S3 Multi-Region Access Points.

Themen

- Erstellen Multi-Regions-Zugriffspunkten
- Konfigurieren eines Multi-Region Access Point zur Verwendung mit AWS PrivateLink
- Stellen von Anforderungen über einen Multi-Region Access Point

Erstellen Multi-Regions-Zugriffspunkten

Gehen Sie wie folgt vor, um einen Multi-Region Access Point in Amazon S3 zu erstellen:

- Geben Sie den Namen für den Multi-Region Access Point an.
- Wählen Sie einen Bucket in jeder AWS-Region aus, in der Sie Anforderungen für den Multi-Region Access Point verarbeiten möchten.
- Konfigurieren Sie die Einstellungen für Amazon S3 Block Public Access für den Multi-Region Access Point.

Sie geben alle diese Informationen in einer Erstellungsanforderung an, die Amazon S3 asynchron verarbeitet. Amazon S3 bietet ein Token, mit dem Sie den Status der asynchronen Erstellungsanforderung überwachen können.

Beheben Sie Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge von AWS Identity and Access Management Access Analyzer bevor Sie Ihre Richtlinie speichern. IAM Access Analyzer führt Richtlinienprüfungen durch, um Ihre Richtlinie anhand der IAM-Richtliniengrammatik und der bewährten Methoden zu validieren. Diese Prüfungen generieren Ergebnisse und bieten umsetzbare Empfehlungen, die Sie beim Erstellen von Richtlinien unterstützen, die funktionsfähig sind und den bewährten Methoden für Sicherheit entsprechen. Weitere Informationen zum Validieren von Richtlinien mit IAM Access Analyzer finden Sie unter Validierung der IAM-Access-Analyzer-Richtlinien im IAM-Benutzerhandbuch. Eine Liste der Warnungen, Fehler und Vorschläge, die von IAM Access Analyzer zurückgegeben werden, finden Sie unter IAM-Access-Analyzer-Richtlinienprüfungsreferenz.

Wenn Sie die API verwenden, ist die Anforderung zum Erstellen eines Multi-Regions-Zugriffpunkts asynchron. Wenn Sie eine Anforderung zum Erstellen eines Multi-Regions-Zugriffpunkts übermitteln, autorisiert Amazon S3 die Anforderung synchron. Es gibt dann sofort ein Token zurück, mit dem Sie den Fortschritt der Erstellungsanforderung verfolgen können. Weitere Informationen zum Nachverfolgen asynchroner Anforderungen zum Erstellen und Verwalten von Multi-Regions-Zugriffpunkten finden Sie unter Verwenden von Multi-Region Access Points mit unterstützten API-Operationen.

Nachdem Sie den Multi-Regions-Zugriffpunkt erstellt haben, können Sie eine Zugriffssteuerungsrichtlinie für diesen erstellen. Jedem Multi-Regions-Zugriffpunkt kann eine Richtlinie zugeordnet sein. Eine Richtlinie für Multi-Region Access Points ist eine ressourcenbasierte Richtlinie, mit der Sie die Verwendung des Multi-Region Access Point nach Ressource, Benutzer oder anderen Bedingungen einschränken können.

> **Note**
>
> Damit eine Anwendung oder ein Benutzer über einen Multi-Regions Access Point auf ein Objekt zugreifen kann, müssen die folgenden Richtlinien beide die Anforderung zulassen:
>
> - Die Zugriffsrichtlinie für den Multi-Region Access Point
> - Die Zugriffsrichtlinie für den zugrunde liegenden Bucket, der das Objekt enthält
>
> Wenn die beiden Richtlinien unterschiedlich sind, hat die restriktivere Richtlinie Vorrang.

Um die Verwaltung von Berechtigungen für Multi-Region Access Points zu vereinfachen, können Sie die Zugriffssteuerung vom Bucket an den Multi-Region Access Point delegieren. Weitere Informationen finden Sie unter the section called “Beispielrichtlinien für Multi-Region Access Points”.

Durch die Verwendung eines Buckets mit einem Multi-Region Access Point ändert sich das Verhalten des Buckets bei Zugriff auf den Bucket über den vorhandenen Bucket-Namen oder einen Amazon-Ressourcennamen (ARN) nicht. Alle vorhandenen Vorgänge für den Bucket funktionieren weiterhin wie zuvor. Einschränkungen, die Sie in eine Multi-Regions-Zugriffspunktrichtlinie einschließen, gelten nur für Anforderungen, die über den Multi-Regions-Zugriffspunkt eingehen.

Sie können die Richtlinie für einen Multi-Regions-Zugriffpunkt nach dem Erstellen aktualisieren, die Richtlinie jedoch nicht löschen. Sie können jedoch die Richtlinie für Multi-Region Access Points aktualisieren, um alle Berechtigungen zu verweigern.

Themen

Regeln zur Benennung von Amazon S3-Multi-Regions-Zugriffspunkten

Wenn Sie einen Multi-Regions-Zugriffpunkt erstellen, geben Sie ihm einen Namen, bei dem es sich um eine von Ihnen ausgewählte Zeichenfolge handelt. Sie können den Namen des Multi-Regions-Zugriffspunkts nach der Erstellung nicht mehr ändern. Der Name muss in Ihrem AWS-Konto einzigartig sein, und es muss den Namensanforderungen entsprechen, die in Einschränkungen und Beschränkungen des Multi-Regions-Zugriffspunkts aufgelistet sind. Wenn Sie den Multi-Regions-Zugriffpunkt identifizieren möchten, verwenden Sie einen Namen, der für Sie oder Ihre Organisation aussagekräftig ist oder der das Szenario widerspiegelt.

Sie verwenden diesen Namen beim Aufrufen von Sie verwenden diesen Namen, wenn Sie Verwaltungsvorgänge für Multi-Regions-Zugriffpunkten, z. B. `GetMultiRegionAccessPoint`

und `PutMultiRegionAccessPointPolicy`. Der Name wird nicht verwendet, um Anforderungen an den Multi-Region Access Point zu senden, und er muss Clients, die Anforderungen unter Verwendung des Multi-Region Access Point stellen, nicht offengelegt werden.

Wenn Amazon S3 einen Multi-Regions-Zugriffpunkt erstellt, weist er ihm automatisch einen Alias zu. Dieser Alias ist eine eindeutige alphanumerische Zeichenfolge, die mit `.mrap` endet. Der Alias wird verwendet, um den Hostnamen und den Amazon-Ressourcennamen (ARN) für einen Multi-Regions-Zugriffspunkt zu erstellen. Der vollqualifizierte Name basiert auch auf dem Alias für den Multi-Regions-Zugriffpunkt.

Sie können den Namen eines Multi-Regions-Zugriffspunkts nicht anhand seines Alias ermitteln, sodass Sie einen Alias offenlegen können, ohne den Namen, den Zweck oder den Besitzer des Multi-Regions-Zugriffspunkts zu enthüllen. Amazon S3 wählt den Alias für jeden neuen Multi-Regions-Zugriffspunkt aus, und der Alias kann nicht geändert werden. Weitere Informationen zur Adressierung eines Multi-Regions-Zugriffspunkts finden Sie unter Stellen von Anforderungen über einen Multi-Region Access Point.

Multi-Regions-Zugriffspunkt-Aliase sind im Laufe der Zeit eindeutig und basieren nicht auf dem Namen oder der Konfiguration eines Multi-Regions-Zugriffspunkts. Wenn Sie einen Multi-Regions-Zugriffspunkt erstellen, ihn dann löschen und einen anderen mit demselben Namen und derselben Konfiguration erstellen, hat der zweite Multi-Regions-Zugriffspunkt einen anderen Alias als der erste. Neue Multi-Regions-Zugriffspunkte können nie denselben Alias wie ein vorheriger Multi-Regions-Zugriffspunkt haben.

Regeln für die Auswahl von Buckets für Amazon S3-Multi-Regions-Zugriffspunkte

Jeder Multi-Regions-Zugriffspunkt ist den Regionen zugeordnet, in denen Sie Anforderungen erfüllen möchten. Der Multi-Regions-Zugriffspunkt muss genau einem Bucket in jeder dieser Regionen zugeordnet sein. Sie geben den Namen jedes Buckets in der Anforderung an, um den Multi-Regions-Zugriffspunkts zu erstellen. Buckets, die den Multi-Region Access Point unterstützen, können sich entweder in dem AWS-Konto befinden, das Eigentümer des Multi-Region Access Point ist, oder in einem anderen AWS-Konten.

Ein einzelner Bucket kann von mehreren Multi-Regions-Zugriffspunkten verwendet werden.

Important

- Sie können die Buckets, die einem Multi-Regions-Zugriffspunkt zugeordnet sind, nur zum Zeitpunkt seiner Erstellung angeben. Nach der Erstellung können Sie keine Buckets hinzufügen, ändern oder aus der Konfiguration des Multi-Regions-Zugriffspunkts entfernen. Um die Buckets zu ändern, müssen Sie den gesamten Multi-Regions-Zugriffspunkt löschen und einen neuen erstellen.
- Sie können einen Bucket, der Teil eines Multi-Regions-Zugriffspunkts ist, nicht löschen. Wenn Sie einen Bucket löschen möchten, der einem Multi-Region Access Point zugeordnet ist, löschen Sie zuerst den Multi-Region Access Point.
- Wenn Sie Ihrem Multi-Region Access Point einen Bucket hinzufügen, dessen Eigentümer ein anderes Konto ist, muss der Bucket-Eigentümer ebenfalls seine Bucket-Richtlinie aktualisieren, um Zugriffsberechtigungen für den Multi-Region Access Point zu gewähren. Andernfalls kann der Multi-Region Access Point keine Daten aus diesem Bucket abrufen. Beispielrichtlinien, die zeigen, wie ein solcher Zugriff gewährt wird, finden Sie unter Beispielrichtlinien für Multi-Region Access Points.
- Nicht alle Regionen unterstützen Multi-Regions-Zugriffspunkte. Eine Liste der unterstützten Regionen finden Sie unter Einschränkungen und Beschränkungen des Multi-Regions-Zugriffspunkts.

Sie können Replikationsregeln erstellen, um Daten zwischen Buckets zu synchronisieren. Mit diesen Regeln können Sie Daten automatisch aus Quell-Buckets in Ziel-Buckets kopieren. Die Verbindung von Buckets mit einem Multi-Regions-Zugriffspunkt hat keinen Einfluss auf die Funktionsweise der Replikation. Das Konfigurieren der Replikation mit Multi-Regions-Zugriffspunkten wird in einem späteren Abschnitt beschrieben.

Important

Wenn Sie eine Anforderung an einen Multi-Region Access Point stellen, kennt der Multi-Region Access Point die Dateninhalte der Buckets in dem Multi-Region Access Point nicht. Daher kann es sein, dass der Bucket, der die Anforderung erhält, die angeforderten Daten nicht enthält. Um konsistente Datensätze in den Amazon-S3-Buckets zu schaffen, die einem Multi-Region Access Point zugeordnet sind, empfehlen wir Ihnen, S3 Cross-Region Replication (CRR, regionsübergreifende Replikation) zu konfigurieren. Weitere Informationen

finden Sie unter Konfigurieren einer Replikation zur Verwendung mit Multi-Region Access Points.

Erstellen eines Amazon S3-Multi-Region Access Point

Das folgende Beispiel zeigt, wie ein Multi-Region Access Point über die Amazon-S3-Konsole erstellt werden kann.

Verwenden der S3-Konsole

So erstellen Sie einen Multi-Regions-Zugriffspunkt

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie Multi-Region Access Points erstellen aus, um mit der Erstellung Ihres Multi-Region Access Point zu beginnen.
4. Geben Sie auf der Seite Multi-Region Access Point einen Namen für den Multi-Region Access Point in das Feld Name des Multi-Region Access Point ein.
5. Wählen Sie die Buckets aus, die diesem Multi-Region Access Point zugeordnet werden sollen. Sie können Buckets auswählen, die sich in Ihrem Konto befinden, oder Sie können Buckets aus anderen Konten auswählen.

   > **Note**
   >
   > Sie müssen mindestens einen Bucket aus Ihrem Konto oder anderen Konten hinzufügen. Beachten Sie außerdem, dass Multi-Region-Access Points nur einen Bucket pro AWS-Region unterstützen. Daher können Sie nicht zwei Buckets aus derselben Region hinzufügen. AWS-Regionen, die standardmäßig deaktiviert sind, werden nicht unterstützt.

   - Um einen Bucket hinzuzufügen, der sich in Ihrem Konto befindet, wählen Sie Hinzufügen von Buckets aus. Es wird eine Liste aller Buckets in Ihrem Konto angezeigt. Sie können nach Ihrem Bucket anhand des Namens suchen oder die Bucket-Namen in alphabetischer Reihenfolge sortieren.

- Um einen Bucket aus einem anderen Konto hinzuzufügen, wählen Sie Bucket aus einem anderen Konto hinzufügen aus. Stellen Sie sicher, dass Sie den genauen Bucket-Namen und die AWS-Konto-ID kennen, da Sie nicht nach Buckets in anderen Konten suchen bzw. diese nicht durchsuchen können.

> **Note**
>
> Sie müssen eine gültige AWS-Konto-ID und einen gültigen Bucket-Namen eingeben. Der Bucket muss sich auch in einer unterstützten Region befinden, andernfalls tritt ein Fehler auf, wenn Sie versuchen, Ihren Multi-Region Access Point zu erstellen. Eine Liste der Regionen, die Multi-Region Access Points unterstützen, finden Sie unter Einschränkungen und Beschränkungen von Multi-Region Access Points.

6. (Optional) Wenn Sie einen hinzugefügten Bucket entfernen müssen, wählen Sie Entfernen aus.

> **Note**
>
> Nach dem Erstellen des Multi-Region Access Point können Sie diesem keine Buckets hinzufügen und keine Buckets daraus entfernen.

7. Wählen Sie unter Block Public Access settings for this Multi-Region Access Point (Einstellungen für die Sperrung des öffentlichen Zugriffs für diesen Multi-Regions-Zugriffspunkt) die Einstellungen für die Sperrung des öffentlichen Zugriffs, die Sie für den Multi-Regions-Zugriffspunkt anwenden möchten. Alle Block-Einstellungen des öffentlichen Zugriffs sind standardmäßig für neue Multi-Regions-Zugriffspunkte aktiviert. Es wird empfohlen, alle Einstellungen aktiviert zu lassen, es sei denn, Sie wissen, dass Sie eine bestimmte Notwendigkeit haben, eine von ihnen zu deaktivieren.

> **Note**
>
> Nach dem Erstellen eines Multi-Region Access Point können Sie die Block-Public-Access-Einstellungen für den Multi-Region Access Point nicht mehr ändern. Wenn Sie den öffentlichen Zugriff blockieren möchten, stellen Sie daher sicher, dass Ihre Anwendungen ohne öffentlichen Zugriff ordnungsgemäß funktionieren, bevor Sie einen Multi-Region Access Point erstellen.

8. Wählen Sie Multi-Regions-Zugriffspunkt erstellen.

> **Important**
>
> Wenn Sie Ihrem Multi-Region Access Point einen Bucket hinzufügen, dessen Eigentümer ein anderes Konto ist, muss der Bucket-Eigentümer auch seine Bucket-Richtlinie aktualisieren, um Zugriffsberechtigungen für den Multi-Region Access Point zu gewähren. Andernfalls kann der Multi-Region Access Point keine Daten aus diesem Bucket abrufen. Beispielrichtlinien, die zeigen, wie ein solcher Zugriff gewährt wird, finden Sie unter Beispielrichtlinien für Multi-Region Access Points.

Verwendung von AWS CLI

Sie können das AWS CLI nutzen, um einen Multi-Regions-Zugriffspunkt zu erstellen. Beim Erstellen des Multi-Region Access Point müssen Sie alle Buckets bereitstellen, die dieser unterstützen soll. Sie können dem Multi-Region Access Point nach der Erstellung keine Buckets hinzuzufügen.

Im folgenden Beispiel wird ein Multi-Region Access Point mit zwei Buckets über die AWS CLI erstellt. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control create-multi-region-access-point --account-id 111122223333 --details '{
        "Name": "simple-multiregionaccesspoint-with-two-regions",
        "PublicAccessBlock": {
            "BlockPublicAcls": true,
            "IgnorePublicAcls": true,
            "BlockPublicPolicy": true,
            "RestrictPublicBuckets": true
        },
        "Regions": [
            { "Bucket": "DOC-EXAMPLE-BUCKET1" },
            { "Bucket": "DOC-EXAMPLE-BUCKET2" }
        ]
    }' --region us-west-2
```

Blockieren des öffentlichen Zugriffs mit Amazon S3-Multi-Regions-Zugriffspunkten

Jeder Multi-Regions-Zugriffspunkt hat verschiedene Block-Einstellungen des öffentlichen Zugriffs in Amazon S3. Diese Einstellungen funktionieren in Verbindung mit den Block-Public-Access-

Einstellungen für das AWS-Konto, das sowohl Eigentümer des Multi-Region Access Point als auch der zugrunde liegenden Buckets ist.

Wenn Amazon S3 eine Anforderung autorisiert, wird die restriktivste Kombination dieser Einstellungen angewendet. Wenn die Block-Public-Access-Einstellungen für eine dieser Ressourcen (das Konto, das Eigentümer des Multi-Region Access Point ist, der zugrunde liegende Bucket oder das Bucket-Eigentümerkonto) den Zugriff für die angeforderte Aktion oder Ressource blockieren, lehnt Amazon S3 die Anforderung ab.

Alle Block-Einstellungen des öffentlichen Zugriffs sind standardmäßig für neue Zugriffspunkte aktiviert, und wir empfehlen Ihnen, alle Einstellungen aktiviert zu lassen, es sei denn, Sie wissen, dass Sie einen bestimmten Wert deaktivieren müssen. Alle Block-Einstellungen des öffentlichen Zugriffs sind standardmäßig für Multi-Regions-Zugriffspunkte aktiviert. Wenn Block Public Access aktiviert ist, kann der Multi-Region Access Point keine internetbasierten Anforderungen annehmen.

> **Important**
>
> Nach dem Erstellen eines Multi-Region Access Point können Sie die Block-Public-Access-Einstellungen für den Access Point nicht mehr ändern.

Weitere Informationen über Block Public Access in Amazon S3 finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Anzeigen der Konfigurationsdetails für Amazon S3 Multi-Region Access Points

Das folgende Beispiel zeigt, wie Konfigurationsdetails für einen Multi-Region Access Point über die Amazon-S3-Konsole angezeigt werden können.

Verwenden der S3-Konsole

So erstellen Sie einen Multi-Regions-Zugriffspunkt

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie den Namen des Multi-Region Access Point aus, dessen Konfigurationsdetails Sie anzeigen möchten.

- Auf der Registerkarte Eigenschaften sind alle Ihrem Multi-Region-Access Point zugeordneten Buckets, das Erstellungsdatum, der Amazon-Ressourcenname (ARN) und der Alias aufgeführt. In der Spalte für die AWS-Konto-ID sind auch alle Buckets aufgeführt, deren Eigentümer externe Konten sind und die Ihrem Multi-Region-Access Point zugeordnet sind.
- Auf der Registerkarte Berechtigungen sind die Block-Public-Access-Einstellungen aufgeführt, die für die diesem Multi-Regions-Zugriffspunkt zugeordneten Buckets gelten. Sie können sich die Richtlinie für Multi-Region Access Points für Ihren Multi-Region Access Point auch ansehen, falls Sie eine erstellt haben. In der Info-Warnung auf der Seite Berechtigungen sind außerdem alle Buckets (in Ihrem Konto und anderen Konten) für diesen Multi-Region Access Point aufgeführt, für die die Einstellung Öffentlicher Zugriff ist blockiert aktiviert ist.
- Die Registerkarte Replikation und Failover bietet eine Kartenansicht der Buckets, die Ihrem Multi-Region Access Point zugeordnet sind, sowie der Regionen, in denen sich die Buckets befinden. Wenn es Buckets von einem anderen Konto gibt, für das Sie keine Berechtigung zum Abrufen von Daten haben, ist die Region in der Replikationsübersicht rot markiert, um darauf hinzuweisen, dass es sich um eine AWS-Region handelt, bei der Fehler beim Abrufen des Replikationsstatus aufgetreten sind.

  > **Note**
  >
  > Um Informationen zum Replikationsstatus aus einem Bucket in einem externen Konto abzurufen, muss Ihnen der Bucket-Eigentümer in seiner Bucket-Richtlinie die Berechtigung `s3:GetBucketReplication` erteilen.

  Auf dieser Registerkarte sind auch die Replikationsmetriken, Replikationsregeln und Failover-Status für die Regionen aufgeführt, die mit Ihrem Multi-Region Access Point verwendet werden.

Verwendung von AWS CLI

Sie können die AWS CLI verwenden, um die Konfigurationsdetails für einen Multi-Region Access Point anzuzeigen.

Im folgenden AWS CLI-Beispiel wird die aktuelle Konfiguration Ihres Multi-Region Access Point abgerufen. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control get-multi-region-access-point --account-id 111122223333 --name DOC-
EXAMPLE-BUCKET1
```

Löschen eines Multi-Region Access Point

Das folgende Verfahren zeigt, wie ein Multi-Region Access Point über die Amazon-S3-Konsole gelöscht werden kann.

Beim Löschen eines Multi-Region Access Point werden die dem Multi-Region Access Point zugeordneten Buckets nicht gelöscht, nur der Multi-Region Access Point selbst.

Verwenden der S3-Konsole

So löschen Sie einen Multi-Region Access Point

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie das Optionsfeld neben dem Namen Ihres Multi-Region Access Point aus.
4. Wählen Sie Delete (Löschen).
5. Geben Sie im Dialogfeld Multi-Region Access Point löschen den Namen des AWS-Buckets ein, den Sie löschen möchten.

   > **Note**
   >
   > Stellen Sie sicher, dass Sie einen gültigen Bucket-Namen eingeben. Andernfalls wird die Schaltfläche Löschen deaktiviert.

6. Wählen Sie Löschen aus, um zu bestätigen, dass Sie Ihren Multi-Region Access Point tatsächlich löschen möchten.

Verwendung von AWS CLI

Sie können die AWS CLI verwenden, um einen Multi-Region Access Point zu löschen. Durch diese Aktion werden die dem Multi-Region Access Point zugeordneten Buckets nicht gelöscht, nur der Multi-Region Access Point selbst. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control delete-multi-region-access-point --account-id 123456789012 --details
 Name=example-multi-region-access-point-name
```

Konfigurieren eines Multi-Region Access Point zur Verwendung mit AWS PrivateLink

Sie können Multi-Region Access Points verwenden, um Amazon-S3-Anforderungsdatenverkehr zwischen AWS-Regionen weiterzuleiten. Jeder globale Endpunkt von Multi-Region Access Points leitet den Amazon-S3-Datenanforderungsverkehr aus mehreren Quellen weiter, ohne dass Sie komplexe Netzwerkkonfigurationen mit separaten Endpunkten erstellen müssen. Zu diesen Quellen des Datenanforderungsverkehrs gehören:

- Datenverkehr aus einer Virtual Private Cloud (VPC)
- Datenverkehr aus On-Premises-Rechenzentren, der über AWS PrivateLink läuft
- Datenverkehr aus dem öffentlichen Internet

Wenn Sie eine AWS PrivateLink-Verbindung zu einem S3 Multi-Region Access Point herstellen, können Sie S3-Anforderungen in AWS oder zwischen mehreren AWS-Regionen über eine private Verbindung weiterleiten, indem Sie eine einfache Netzwerkarchitektur und -konfiguration verwenden. Wenn Sie AWS PrivateLink verwenden, müssen Sie keine VPC-Peering-Verbindung konfigurieren.

Themen

Konfigurieren eines Multi-Regions-Zugriffspunkts zur Verwendung mit AWS PrivateLink

AWS PrivateLink bietet Ihnen eine private Konnektivität mit Amazon S3 über private IP-Adressen in Ihrer Virtual Private Cloud (VPC). Sie können einen oder mehrere Schnittstellenendpunkte in Ihrer VPC bereitstellen, um eine Verbindung zu Amazon S3-Multi-Regions-Zugriffspunkten herzustellen.

Sie können com.amazonaws.s3-global.accesspoint-Endpunkte für Multi-Regions-Zugriffspunkte über die AWS Management Console, AWS CLI, oder AWS-SDKs. erstellen. Weitere Informationen

zum Konfigurieren eines Schnittstellenendpunkts für Multi-Regions-Zugriffspunkte finden Sie unter Schnittstellen-VPC-Endpunkte im VPC-Benutzerhandbuch.

Gehen Sie folgendermaßen vor, um Anforderungen an einen Multi-Regions-Zugriffspunkt über Schnittstellenendpunkte zu senden, um die VPC und den Multi-Regions-Zugriffspunkt zu konfigurieren.

Konfigurieren eines Multi-Regions-Zugriffspunkts zur Verwendung mit AWS PrivateLink

1. Erstellen oder verfügen Sie über einen geeigneten VPC-Endpunkt, der eine Verbindung zu Multi-Regions-Zugriffspunkten herstellen kann. Weitere Informationen zum Erstellen von VPC-Endpunkten finden Sie unter Schnittstelle-VPC-Endpunkte im VPC-Benutzerhandbuch.

   > **Important**
   >
   > Stellen Sie sicher, dass Sie einen com.amazonaws.s3-global.accesspoint-Endpunkt erstellen. Andere Endpunkttypen können nicht auf Multi-RegionsZugriffspunkte zugreifen.

   Nachdem dieser VPC-Endpunkt erstellt wurde, leiten alle Multi-Regions-Zugriffspunkt-Anforderungen in der VPC durch diesen Endpunkt, wenn Sie private DNS für den Endpunkt aktiviert haben. Dies ist standardmäßig aktiviert.
2. Wenn die Multi-Regions-Zugriffspunkt-Richtlinie keine Verbindungen von VPC-Endpunkten unterstützt, müssen Sie sie aktualisieren.
3. Stellen Sie sicher, dass die einzelnen Bucket-Richtlinien den Zugriff auf die Benutzer des Multi-Regions-Zugriffspunkts ermöglichen.

Denken Sie daran, dass Multi-Regions-Zugriffspunkte funktionieren, indem sie Anforderungen an Buckets weiterleiten, nicht indem sie selbst Anforderungen erfüllen. Dies ist wichtig, da der Absender der Anforderung über Berechtigungen für den Multi-Regions-Zugriffspunkt verfügen muss und auf die einzelnen Buckets im Multi-Regions-Zugriffspunkt zugreifen kann. Andernfalls wird die Anforderung möglicherweise an einen Bucket weitergeleitet, in dem der Originator keine Berechtigungen hat, um die Anforderung zu erfüllen. Ein Multi-Region Access Point und die zugeordneten Buckets können im Besitz desselben oder eines anderen AWS-Kontos sein. VPCs aus verschiedenen Konten können jedoch einen Multi-Regions-Zugriffspunkt verwenden, wenn die Berechtigungen korrekt konfiguriert sind.

Aus diesem Grund muss die VPC-Endpunktrichtlinie sowohl Zugriff auf den Multi-Regions-Zugriffspunkt als auch auf jeden zugrunde liegenden Bucket gewähren, der Anforderungen erfüllen soll. Angenommen, Sie haben einen Multi-Region Access Point mit dem Alias `mfzwi23gnjvgw.mrap`. Es wird von Buckets `DOC-EXAMPLE-BUCKET1` und `DOC-EXAMPLE-BUCKET2` gesichert, die alle im Besitz von AWS-Konto `123456789012` sind. In diesem Fall würde die folgende VPCE-Endpunktrichtlinie zulassen, dass `GetObject`-Anforderungen von der VPC an `mfzwi23gnjvgw.mrap` von einem der beiden Backing-Buckets erfüllt werden.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Sid": "Read-buckets-and-MRAP-VPCE-policy",
        "Principal": "*",
        "Action": [
            "s3:GetObject"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*",
            "arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap/object/*"
        ]
    }]
}
```

Wie bereits erwähnt, müssen Sie auch sicherstellen, dass die Multi-Regions-Zugriffspunkt-Richtlinie so konfiguriert ist, dass der Zugriff über einen VPC-Endpunkt unterstützt wird. Sie müssen den VPC-Endpunkt, der den Zugriff anfordert, nicht angeben. Die folgende Beispielrichtlinie würde jedem Anforderer Zugriff gewähren, der versucht, den Multi-Region Access Point für die `GetObject`-Anforderungen zu nutzen.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Sid": "Open-read-MRAP-policy",
        "Effect": "Allow",
        "Principal": "*",
        "Action": [
            "s3:GetObject"
```

```
          ],
        "Resource": "arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap/object/*"
    }]
}
```

Und natürlich würden die einzelnen Buckets jeweils eine Richtlinie benötigen, um den Zugriff von Anforderungen zu unterstützen, die über den VPC-Endpunkt gesendet werden. Die folgende Beispielrichtlinie gewährt Lesezugriff auf alle anonymen Benutzer, einschließlich Anforderungen, die über den VPC-Endpunkt gestellt werden.

```
{
    "Version":"2012-10-17",
   "Statement": [
   {
       "Sid": "Public-read",
       "Effect":"Allow",
       "Principal": "*",
       "Action": "s3:GetObject",
       "Resource": [
           "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
           "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*"]
    }]
}
```

Weitere Informationen zur Bearbeitung einer VPC-Endpunktrichtlinie finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im VPC-Benutzerhandbuch.

Entfernen des Zugriffs auf einen Multi-Region Access Point von einem VPC-Endpunkt

Wenn Sie einen Multi-Region Access Point besitzen und den Zugriff darauf von einem Schnittstellenendpunkt entfernen möchten, müssen Sie eine neue Zugriffsrichtlinie für den Multi-Region Access Point bereitstellen, die den Zugriff für Anforderungen über VPC-Endpunkte verhindert. Wenn die Buckets in Ihrem Multi-Region Access Point Anforderungen über VPC-Endpunkte unterstützen, werden sie diese Anforderungen allerdings weiterhin unterstützen. Wenn Sie diesen Support verhindern möchten, müssen Sie auch die Richtlinien für die Buckets aktualisieren. Durch die Bereitstellung einer neuen Zugriffsrichtlinie für den Multi-Region Access Point wird nur der Zugriff auf den Multi-Region Access Point verhindert, nicht auf die zugrunde liegenden Buckets.

> **Note**
>
> Sie können eine Zugriffsrichtlinie für einen Multi-Regions-Zugriffspunkt nicht löschen. Um den Zugriff auf einen Multi-Regions-Zugriffspunkt zu entfernen, müssen Sie eine neue Zugriffsrichtlinie mit dem gewünschten geänderten Zugriff bereitstellen.

Anstatt die Zugriffsrichtlinie für den Multi-Region Access Point zu aktualisieren, können Sie die Bucket-Richtlinien aktualisieren, um Anforderungen über VPC-Endpunkte zu verhindern. In diesem Fall können Benutzer weiterhin über den VPC-Endpunkt auf den Multi-Region Access Point zugreifen. Wenn der Multi-Region Access Point jedoch an einen Bucket weitergeleitet wird, in dem die Bucket-Richtlinie den Zugriff verhindert, wird eine Fehlermeldung generiert.

Stellen von Anforderungen über einen Multi-Region Access Point

Wie andere Ressourcen verfügen auch Amazon S3 Multi-Region Access Points über Amazon-Ressourcennamen (ARNs). Sie können diese ARNs verwenden, um Anforderungen über die AWS Command Line Interface (AWS CLI), AWS-SDKs oder die Amazon-S3-API an Multi-Region Access Points zu leiten. Sie können diese ARNs auch verwenden, um Multi-Region Access Points in Zugriffskontrollrichtlinien zu identifizieren. Der ARN eines Multi-Region Access Point beinhaltet den Namen des Multi-Region Access Points nicht und gibt diesen nicht an. Weitere Informationen zur Verwendung von ARNs finden Sie unter Amazon-Ressourcennamen (ARN) im Allgemeine AWS-Referenz.

> **Note**
>
> Der Alias des Multi-Region Access Point und der ARN können nicht gegeneinander ausgetauscht werden.

ARNs von Multi-Region Access Points verwenden das folgende Format:

`arn:aws:s3::`*`account-id`*`:accesspoint/`*`MultiRegionAccessPoint_alias`*

Im Folgenden finden Sie einige Beispiele für ARNs von Multi-Region Access Points:

- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap` repräsentiert den Multi-Region Access Point mit dem Alias `mfzwi23gnjvgw.mrap`, der im Besitz von AWS-Konto `123456789012` ist.

- `arn:aws:s3::123456789012:accesspoint/*` repräsentiert alle Multi-Region Access Points im Konto `123456789012`. Dieser ARN stimmt mit allen Multi-Region Access Points für das Konto `123456789012` überein, entspricht aber keinen regionalen Amazon S3 Access Points, da der ARN keine AWS-Region enthält. Im Gegensatz dazu stimmt der ARN `arn:aws:s3:us-west-2:123456789012:accesspoint/*` mit allen regionalen Amazon S3 Access Points in der Region `us-west-2` für das Konto `123456789012` überein, entspricht aber keinem Multi-Region Access Point.

ARNs für Objekte, auf die über einen Multi-Region Access Point zugegriffen wird, verwenden folgendes Format:

`arn:aws:s3::`*`account_id`*`:accesspoint/`*`MultiRegionAccessPoint_alias`*`//`*`key`*

Wie bei ARNs von Multi-Region Access Points enthalten die ARNs für Objekte, auf die über Multi-Region Access Points zugegriffen wird, keine AWS-Region. Hier sind einige Beispiele.

- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap//-01` repräsentiert `-01`, auf das über den Multi-Region Access Point mit dem Alias `mfzwi23gnjvgw.mrap`, im Besitz von Konto `123456789012`, zugegriffen wird.
- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap//*` repräsentiert alle Objekte, auf die über den Multi-Region Access Point mit dem Alias `mfzwi23gnjvgw.mrap`, in Konto `123456789012`, zugegriffen werden kann.
- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap//-01/finance/*` repräsentiert alle Objekte, auf die unter `-01/finance/` für den Multi-Region Access Point mit dem Alias `mfzwi23gnjvgw.mrap`, in Konto `123456789012`, zugegriffen werden kann.

Hostnamen für Multi-Regions-Zugriffspunkte

Sie können über einen Multi-Region Access Point auf Daten in Amazon S3 zugreifen, indem Sie den Hostnamen des Multi-Region Access Points verwenden. Anforderungen können aus dem öffentlichen Internet an diesen Hostnamen gerichtet werden. Wenn Sie ein oder mehrere Internet-Gateways für den Multi-Region Access Point konfiguriert haben, können Anforderungen auch aus einer Virtual Private Cloud (VPC) an diesen Hostnamen weitergeleitet werden. Weitere Informationen zum Erstellen von VPC-Schnittstellenendpunkten zur Verwendung mit mehreren Multi-Regions-Zugriffspunkten finden Sie unter Konfigurieren eines Multi-Regions-Zugriffspunkts zur Verwendung mit AWS PrivateLink.

Wenn Sie Anforderungen über einen Multi-Region Access Point von einer VPC unter Verwendung eines VPC-Endpunkts stellen möchten, können Sie AWS PrivateLink verwenden. Wenn Sie Anforderungen an einen Multi-Region Access Point mithilfe von AWS PrivateLink stellen, können Sie den Namen eines endpunktspezifischen regionalen DOMAIN NAME SYSTEM (DNS), der mit *region*`.vpce.amazonaws.com` endet, nicht direkt verwenden. Diesem Hostnamen wird kein Zertifikat zugeordnet, so dass er nicht direkt verwendet werden kann. Sie können den DOMAIN NAME SYSTEM (DNS)-Namen des VPC-Endpunkts weiterhin als `CNAME`- oder `ALIAS`-Ziel verwenden. Alternativ können Sie privates DOMAIN NAME SYSTEM (DNS) auf dem Endpunkt aktivieren und den standardmäßigen DNS-Namen *MultiRegionAccessPoint_alias*`.accesspoint.s3-global.amazonaws.com` des Multi-Region Access Point wie in diesem Abschnitt beschrieben verwenden.

Wenn Sie Anforderungen an die API für Amazon-S3-Datenoperationen (z. B. `GetObject`) über einen Multi-Region Access Point stellen, lautet der Hostname für die Anforderung wie folgt:

MultiRegionAccessPoint_alias`.accesspoint.s3-global.amazonaws.com`

Wenn Sie zum Beispiel eine `GetObject`-Anforderung über den Multi-Region Access Point mit dem Alias `mfzwi23gnjvgw.mrap` stellen möchten, stellen Sie eine Anforderung an den Hostnamen `mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com`. Der `s3-global`-Teil des Hostnamens gibt an, dass dieser Hostname nicht für eine bestimmte Region bestimmt ist.

Das Senden von Anforderungen über einen Multi-Regions-Zugriffspunkt ähnelt der Ausführung von Anforderungen über einen Zugriffspunkt für einzelne Regionen. Es ist jedoch wichtig, folgende Unterschiede zu beachten:

- ARNs von Multi-Region Access Points enthalten keine AWS-Region. Sie folgen dem Format `arn:aws:s3::`*account-id*`:accesspoint/`*MultiRegionAccessPoint_alias*.
- Für Anforderungen, die über API-Operationen gestellt werden (diese Anforderungen erfordern keinen ARN), verwenden Multi-Region Access Points ein anderes Endpunktschema. Das Schema ist *MultiRegionAccessPoint_alias*`.accesspoint.s3-global.amazonaws.com` – z. B. `mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com`. Beachten Sie die Unterschiede im Vergleich zu einem Zugriffspunkt für einzelne Regionen:
  - Hostnamen des Multi-Regions-Zugriffspunkts verwenden ihren Alias, nicht den Namen des Multi-Regions-Zugriffspunkts.
  - Hostnamen von Multi-Region Access Points enthalten nicht die AWS-Konto-ID des Besitzers.
  - Hostnamen von Multi-Region Access Points enthalten keine AWS-Region.

  - Zu den Multi-Regions-Hostnamen gehören `s3-global.amazonaws.com` anstelle von `s3.amazonaws.com`.
- Anforderungen für Multi-Region Access Points müssen mit Signature Version 4A (SigV4A) signiert werden. Wenn Sie die AWS-SDKs verwenden, konvertiert das SDK ein SigV4 automatisch in SigV4A. Stellen Sie daher sicher, dass Ihr AWS-SDK SigV4A als Signaturimplementierung unterstützt, die zum Signieren der globalen Anforderungen der AWS-Region verwendet wird. Weitere Informationen zu SigV4A finden Sie unter Signieren von AWS-API-Anforderungen in der Allgemeine AWS-Referenz.

Multi-Regions-Zugriffspunkte und Amazon S3 Transfer Acceleration

Amazon S3 Transfer Acceleration ist eine Funktion, die schnelle Datenübertragungen zu Buckets ermöglicht. Transfer Acceleration wird auf der Ebene des einzelnen Buckets konfiguriert. Weitere Informationen zu Transfer Acceleration finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.

Multi-Region Access Points verwenden einen ähnlichen beschleunigten Übertragungsmechanismus wie Transfer Acceleration, um große Objekte über das AWS-Netzwerk zu senden. Aus diesem Grund müssen Sie Transfer Acceleration nicht verwenden, wenn Sie Anforderungen über einen Multi-Region Access Point senden. Diese erhöhte Übertragungsleistung wird automatisch in den Multi-Region Access Point integriert.

Themen

- Berechtigungen
- Einschränkungen und Beschränkungen des Multi-Regions-Zugriffspunkts
- Weiterleitung der Multi-Regions-Zugriffspunktanforderung
- Failover-Kontrollen für Amazon S3 Multi-Region Access Points
- Konfigurieren einer Replikation zur Verwendung mit Multi-Region Access Points
- Verwenden von Multi-Region Access Points mit unterstützten API-Operationen
- Überwachung und Protokollierung von Anforderungen, die über einen Multi-Regions-Zugriffspunkt an zugrunde liegende Ressourcen erfolgen

Berechtigungen

Amazon S3 Multi-Region Access Points können den Datenzugriff für Amazon-S3-Buckets in mehreren AWS-Regionen vereinfachen. Multi-Region Access Points sind benannte globale Endpunkte, mit denen Sie Datenzugriffsoperationen für Objekte in Amazon S3 ausführen können, z. B. `GetObject` und `PutObject`. Jeder Multi-Region Access Point kann für jede Anforderung, die über den globalen Endpunkt eingehen, über unterschiedliche Berechtigungen und Netzwerkkontrollen verfügen.

Jeder Multi-Region Access Point kann außerdem eine benutzerdefinierte Zugriffsrichtlinie erzwingen, die in Verbindung mit der Bucket-Richtlinie funktioniert, welche dem zugrunde liegenden Bucket angefügt ist. Damit eine Anforderung erfolgreich ist, müssen alle folgenden Komponenten die Operation zulassen:

- Die Richtlinie der Multi-Region Access Points
- Die zugrunde liegendeAWS Identity and Access Management (IAM)-Richtlinie
- Die zugrunde liegende Bucket-Richtlinie (an die die Anforderung weitergeleitet wird)

Sie können jede Richtlinie für Multi-Region Access Points so konfigurieren, dass nur Anforderungen von bestimmten IAM-Benutzern oder -Gruppen akzeptiert werden. Ein Beispiel für diese Vorgehensweise finden Sie in Beispiel 2 unter the section called “Beispielrichtlinien für Multi-Region Access Points”. Um den Amazon-S3-Datenzugriff auf ein privates Netzwerk zu beschränken, können Sie die Richtlinie für Multi-Region Access Points auch so konfigurieren, dass Anforderungen nur von einer Virtual Private Cloud (VPC) akzeptiert werden.

Nehmen wir zum Beispiel an, dass Sie eine `GetObject`-Anforderung über einen Multi-Region Access Point mithilfe eines Benutzers namens `AppDataReader` in Ihrem AWS-Konto stellen. Um sicherzustellen, dass die Anforderung nicht abgelehnt wird, muss der Benutzer `AppDataReader` die `s3:GetObject`-Berechtigung vom Multi-Region Access Point und von jedem Bucket, der dem Multi-Region Access Point zugrunde liegt, erhalten. `AppDataReader` kann keine Daten aus einem Bucket abrufen, der diese Berechtigung nicht erteilt.

> **Important**
>
> Das Delegieren der Zugriffskontrolle für einen Bucket an eine Richtlinie eines Multi-Region Access Points ändert nicht das Verhalten des Buckets, wenn auf den Bucket über den Bucket-Namen oder den Amazon-Ressourcennamen (ARN) zugegriffen wird. Alle

Operationen, die direkt für den Bucket ausgeführt werden, funktionieren weiterhin wie zuvor. Einschränkungen, die Sie in eine Richtlinie für Multi-Region Access Points einschließen, gelten nur für Anforderungen, die über diesen Multi-Region Access Point eingehen.

Verwalten des öffentlichen Zugriffs auf einen Multi-Regions-Zugriffspunkt

Multi-Regions-Zugriffspunkte unterstützen unabhängige Block-Einstellungen des öffentlichen Zugriffs für jeden Multi-Regions-Zugriffspunkt. Wenn Sie einen Multi-Regions-Zugriffspunkt erstellen, können Sie die Block-Einstellungen des öffentlichen Zugriffs für diesen Multi-Regions–Zugriffspunkt festlegen.

Note

Alle Einstellungen für „Öffentlichen Zugriff blockieren“, die unter „Einstellungen „Öffentlichen Zugriff beschränken“ für dieses Konto (in Ihrem eigenen Konto) oder Einstellungen „Öffentlichen Zugriff beschränken“ für externe Buckets aktiviert sind, gelten auch dann, wenn die unabhängigen Einstellungen für „Öffentlichen Zugriff blockieren“ für Ihren Multi-Region Access Point deaktiviert sind.

Für jede Anforderung, die über einen Multi-Region Access Point erfolgt, wertet Amazon S3 die Einstellungen zum Blockieren des öffentlichen Zugriffs für Folgendes aus:

- Den Multi-Region Access Point
- Die zugrunde liegenden Buckets (einschließlich externer Buckets)
- Das Konto, das den Multi-Region Access Point besitzt
- Das Konto, das die zugrunde liegenden Buckets besitzt (einschließlich externer Konten)

Wenn eine dieser Einstellungen darauf hinweist, dass die Anforderung gesperrt werden soll, lehnt Amazon S3 die Anforderung ab. Weitere Hinweise zur Funktion Amazon S3 Block Public Access finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Important

Alle Einstellungen für die Blockierung des öffentlichen Zugriffs sind für neue Multi-Region Access Points standardmäßig aktiviert. Sie müssen alle Einstellungen, die Sie nicht auf einen Multi-Regions-Zugriffspunkt anwenden möchten, explizit deaktivieren.

Nach dem Erstellen eines Multi-Region Access Point können Sie die Block-Public-Access-Einstellungen für den Access Point nicht mehr ändern.

Anzeigen der Einstellungen für die Blockierung des öffentlichen Zugriffs für einen Multi-Region Access Point

So zeigen Sie die Einstellungen für die Blockierung des öffentlichen Zugriffs für einen Multi-Region Access Point an

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie den Namen des Multi-Region Access Points aus, den Sie überprüfen möchten.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen).
5. Überprüfen Sie unter Block Public Access settings for this Multi-Region Access Point (Einstellungen für die Blockierung des öffentlichen Zugriffs für diesen Multi-Region Access Point) die Einstellungen für die Blockierung des öffentlichen Zugriffs für Ihren Multi-Region Access Point.

> **Note**
>
> Nachdem der Multi-Region Access Point erstellt wurde, können Sie die Einstellungen für die Blockierung des öffentlichen Zugriffs nicht mehr bearbeiten. Wenn Sie den öffentlichen Zugriff blockieren möchten, stellen Sie daher sicher, dass Ihre Anwendungen ohne öffentlichen Zugriff ordnungsgemäß funktionieren, bevor Sie einen Multi-Region Access Point erstellen.

Verwenden einer Richtlinie für Multi-Region Access Points

Das folgende Beispiel einer Richtlinie für Multi-Region Access Points gewährt einem IAM-Benutzer Zugriff zum Auflisten und Herunterladen von Dateien aus Ihrem Multi-Region Access Point. Wenn Sie diese Beispielrichtlinie verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen.

```
 {
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::123456789012:user/JohnDoe"
         },
         "Action":[
            "s3:ListBucket",
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias",
            "arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias/object/
*"
         ]
      }
   ]
}
```

Verwenden Sie den folgenden Befehl `put-multi-region-access-point-policy`, um die Richtlinie für Multi-Region Access Points über die AWS Command Line Interface (AWS CLI) mit dem angegebenen Multi-Region Access Point zu verknüpfen. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen. Jeder Multi-Region Access Point kann nur über eine Richtlinie verfügen. Daher ersetzt eine an Aktion `put-multi-region-access-point-policy` gerichtete Anforderung jede vorhandene Richtlinie, die dem angegebenen Multi-Region Access Point zugeordnet ist.

AWS CLI

```
aws s3control put-multi-region-access-point-policy
--account-id 111122223333
--details { "Name": "DOC-EXAMPLE-BUCKET-MultiRegionAccessPoint",
 "Policy": "{ \"Version\": \"2012-10-17\", \"Statement\": { \"Effect\":
 \"Allow\", \"Principal\": { \"AWS\": \"arn:aws:iam::111122223333:root
\" }, \"Action\": [\"s3:ListBucket\", \"s3:GetObject\"], \"Resource\":
 [ \"arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias",
 \"arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias/object/*
\" ] } }" }
```

Verwenden Sie den folgenden Befehl, um die Ergebnisse der vorherigen Operation abzufragen:

AWS CLI

```
aws s3control describe-multi-region-access-point-operation
--account-id 111122223333
--request-token-arn requestArn
```

Verwenden Sie den folgenden Befehl, um Ihre Richtlinie für Multi-Region Access Points abzurufen:

AWS CLI

```
aws s3control get-multi-region-access-point-policy
--account-id 111122223333
--name=DOC-EXAMPLE-BUCKET-MultiRegionAccessPoint
```

Bearbeiten der Richtlinie für Multi-Region Access Points

Die Richtlinie für Multi-Region Access Points (in JSON geschrieben) bietet Speicherzugriff auf die Amazon-S3-Buckets, die mit diesem Multi-Region Access Point verwendet werden. Sie können bestimmten Prinzipalen die Ausführung verschiedener Aktionen auf Ihrem Multi-Region Access Point erlauben oder verweigern. Wenn eine Anforderung über den Multi-Region Access Point an einen Bucket weitergeleitet wird, gelten sowohl die Zugriffsrichtlinien für den Multi-Region Access Point als auch den Bucket. Die restriktivere Zugriffsrichtlinie hat immer Vorrang.

> **Note**
>
> Wenn ein Bucket Objekte enthält, die anderen Konten gehören, gilt die Richtlinie für Multi-Region Access Points nicht für Objekte, die anderen AWS-Konten gehören.

Nachdem Sie eine Richtlinie für Multi-Region Access Points angewendet haben, kann die Richtlinie nicht mehr gelöscht werden. Sie können entweder die Richtlinie bearbeiten oder eine neue Richtlinie erstellen, die die bestehende überschreibt.

So bearbeiten Sie die Richtlinie für Multi-Region Access Points

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie den Namen des Multi-Region Access Point aus, für den Sie die Richtlinie bearbeiten möchten.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen).
5. Scrollen Sie nach unten zum Abschnitt Richtlinie für Multi-Region Access Points. Wählen Sie Edit (Bearbeiten) aus, um die Richtlinie (in JSON) zu bearbeiten.
6. Die Seite Edit Multi-Region Access Point policy (Richtlinie für Multi-Region Access Points bearbeiten) wird angezeigt. Sie können die Richtlinie entweder direkt in das Textfeld eingeben oder Add statement (Anweisung hinzufügen) auswählen, um Richtlinienelemente aus einer Dropdownliste auszuwählen.

> **Note**
>
> Die Konsole zeigt automatisch den Amazon-Ressourcennamen (ARN) für den Multi-Region Access Point an, den Sie in der Richtlinie verwenden können. Für Beispielrichtlinien für Multi-Region Access Points vgl. the section called “Beispielrichtlinien für Multi-Region Access Points”.

Beispielrichtlinien für Multi-Region Access Points

Amazon S3 Multi-Region Access Points unterstützen AWS Identity and Access Management (IAM)-Ressourcenrichtlinien. Mithilfe dieser Richtlinien können Sie die Verwendung des Multi-Region Access Point nach Ressource, Benutzer oder anderen Bedingungen steuern. Damit eine Anwendung oder ein Benutzer über einen Multi-Region Access Point auf Objekte zugreifen kann, müssen sowohl der Multi-Region Access Point Zugriffspunkt als auch der zugrunde liegende Bucket den gleichen Zugriff erlauben.

Gehen Sie folgendermaßen vor, um den gleichen Zugriff sowohl auf den Multi-Region Access Point als auch auf den zugrunde liegenden Bucket zu erlauben:

- (Empfohlen) Wenn Sie die Zugriffskontrollen bei der Verwendung eines Amazon S3 Multi-Region Access Points vereinfachen möchten, delegieren Sie die Zugriffskontrolle für den Amazon-S3-Bucket an den Multi-Region Access Point. Ein Beispiel für diese Vorgehensweise finden Sie in Beispiel 1 in diesem Abschnitt.

- Fügen Sie der Richtlinie des zugrunde liegenden Buckets dieselben Berechtigungen hinzu, die in der Richtlinie des Multi Region Access Points enthalten sind.

> **Important**
>
> Das Delegieren der Zugriffskontrolle für einen Bucket an eine Richtlinie eines Multi-Region Access Points ändert nicht das Verhalten des Buckets, wenn auf den Bucket über den Bucket-Namen oder den Amazon-Ressourcennamen (ARN) zugegriffen wird. Alle Operationen, die direkt für den Bucket ausgeführt werden, funktionieren weiterhin wie zuvor. Einschränkungen, die Sie in eine Richtlinie für Multi-Region Access Points einschließen, gelten nur für Anforderungen, die über diesen Multi-Region Access Point eingehen.

Example 1 – Delegieren des Zugriffs auf bestimmte Multi-Region Access Points in Ihrer Bucket-Richtlinie (für dasselbe Konto oder kontoübergreifend)

Das folgende Beispiel für eine Bucket-Richtlinie gewährt vollständigen Bucket-Zugriff auf einen bestimmten Multi-Region Access Point. Dies bedeutet, dass der gesamte Zugriff auf diesen Bucket durch die Richtlinien gesteuert wird, die dem Multi-Region Access Point angefügt sind. Wir empfehlen, Ihre Buckets auf diese Weise für alle Anwendungsfälle zu konfigurieren, die keinen direkten Zugriff auf den Bucket erfordern. Sie können diese Bucket-Richtlinienstruktur für Multi-Region Access Points in demselben Konto oder einem anderen Konto verwenden.

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS": "*" },
        "Action" : "*",
        "Resource" : [ "Bucket ARN", "Bucket ARN/*"],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointArn" : "MultiRegionAccessPoint_ARN" }
        }
    }]
}
```

> **Note**
>
> Wenn Sie für mehrere Multi-Region Access Points Zugriff erteilen, stellen Sie sicher, dass Sie jeden einzelnen Multi-Region Access Point auflisten.

Example 2 – Gewähren des Zugriffs auf einen Multi-Region Access Point für ein Konto in Ihrer Richtlinie für Multi-Region Access Points

Die folgende Richtlinie für Multi-Region Access Points gewährt dem Konto *123456789012* die Berechtigung zum Auflisten und Lesen der Objekte, die in dem Multi-Region Access Point enthalten sind, der durch den *MultiRegionAccessPoint_ARN* definiert wird.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
       "Effect": "Allow",
       "Principal": {
          "AWS":"arn:aws:iam::123456789012:user/JohnDoe"
       },
       "Action":[
          "s3:ListBucket",
          "s3:GetObject"
       ],
       "Resource":[
          "MultiRegionAccessPoint_ARN",
          "MultiRegionAccessPoint_ARN/object/*"
       ]
     }
  ]
}
```

Example 3 – Richtlinie für Multi-Region Access Points, die eine Bucket-Auflistung ermöglicht

Die folgende Richtlinie für Multi-Region Access Points gewährt dem Konto *123456789012* die Berechtigung zum Auflisten der Objekte, die in dem Multi-Region Access Point enthalten sind, der durch den *MultiRegionAccessPoint_ARN* definiert wird.

```
{
   "Version":"2012-10-17",
```

```
   "Statement": [
      {
       "Effect": "Allow",
       "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/JohnDoe"
        },
        "Action": "s3:ListBucket",
        "Resource": "MultiRegionAccessPoint_ARN"
       }
    ]
}
```

Einschränkungen und Beschränkungen des Multi-Regions-Zugriffspunkts

Multi-Regions-Zugriffspunkte in Amazon S3 haben die folgenden Einschränkungen und Beschränkungen.

- Namen für Multi-Regions-Zugriffspunkte:
  - Muss innerhalb eines einzelnen AWS-Kontos eindeutig sein
  - Muss mit einer Zahl oder einem Kleinbuchstaben beginnen
  - Muss zwischen 3 und 50 Zeichen lang sein.
  - Dürfen nicht mit einem Bindestrich (`-`) beginnen oder enden
  - Dürfen keine Unterstriche (`_`), Großbuchstaben oder Punkte (`.`) enthalten
  - Können nicht mehr bearbeitet werden, nachdem sie erstellt wurden
- Aliase für Multi-Region Access Points werden von Amazon S3 generiert und können nicht bearbeitet oder wiederverwendet werden.
- Sie können über einen Multi-Region Access Point nicht mit Gateway-Endpunkten auf Daten zugreifen. Sie können jedoch über einen Multi-Region Access Point mit Schnittstellenendpunkten auf Daten zugreifen. Zur Verwendung von AWS PrivateLink müssen Sie VPC-Endpunkte erstellen. Weitere Informationen finden Sie unter Konfigurieren eines Multi-Regions-Zugriffspunkts zur Verwendung mit AWS PrivateLink.
- Um die Verwaltung von Zugriffspunkten für mehrere Regionen mit Amazon CloudFront zu erleichtern, müssen Sie den Zugriffspunkt für mehrere Regionen als einen `Custom Origin`-Verteilungstyp konfigurieren. Weitere Informationen zu verschiedenen Herkunftstypen finden Sie unter Verwenden verschiedener Ursprünge mit CloudFront-Verteilungen. Weitere Informationen zur Verwendung von Zugriffspunkten für mehrere Regionen mit Amazon CloudFront finden Sie

unter Building an active-active, latency-based application across multiple Regions (Erstellen einer latenzbasierten Aktiv/Aktiv-Anwendung über mehrere Regionen hinweg) im AWS Storage Blog.

- Mindestanforderungen für Multi-Regions-Zugriffspunkte:
  - Transport Layer Security (TLS) v1.2
  - Signatur-Version 4 (SigV4A)

    Multi-Regions-Zugriffspunkte unterstützen Signatur-Version 4A. Diese Version von SigV4 ermöglicht das Signieren von Anforderungen für mehrere AWS-Regionen. Dies ist nützlich bei API-Operationen, die zu Datenzugriff von einer oder mehreren Regionen führen können. Bei Verwendung eines AWS SDK geben Sie Ihre Anmeldeinformationen an und die Anforderungen an Multi-Region Access Points verwenden Signature-Version 4A ohne zusätzliche Konfiguration. Stellen Sie sicher, dass Sie die Kompatibilität Ihres AWS SDK mit dem SigV4A-Algorithmus überprüfen. Weitere Informationen zu SigV4A finden Sie unter Signieren von AWS-API-Anforderungen in der Allgemeine AWS-Referenz.

    > **Note**
    >
    > Stellen Sie bei der Verwendung von SigV4A mit temporären Sicherheitsanmeldeinformationen, z. B. bei der Verwendung von AWS Identity and Access Management (IAM)-Rollen, sicher, dass Sie die temporären Anmeldeinformationen nicht von einem globalen Endpunkt, sondern von einem regionalen Endpunkt in AWS Security Token Service (AWS STS) anfordern. Wenn Sie den globalen Endpunkt für AWS STS (`sts.amazonaws.com`) verwenden, generiert AWS STS temporäre Anmeldeinformationen von einem globalen Endpunkt, der von Sig4A nicht unterstützt wird. Aus diesem Grund erhalten Sie eine Fehlermeldung. Verwenden Sie einen der aufgelisteten regionalen Endpunkte für AWS STS, um dieses Problem zu beheben.

- Multi-Region Access Points unterstützen keine anonymen Anforderungen.
- Beschränkungen des Multi-Regions-Zugriffspunkts:
  - IPv6 wird nicht unterstützt.
  - Buckets von Amazon S3 on Outposts werden nicht unterstützt.
  - `CopyObject` wird nicht unterstützt, weder als Quelle noch als Ziel.
  - Die Funktion S3 Batch Operations wird nicht unterstützt.

- Bestimmte AWS SDKs werden nicht unterstützt. Informationen dazu, welche AWS SDKs für Multi-Region Access Points unterstützt werden, finden Sie unter Kompatibilität mit AWS SDKs.
- Service Quotas für Multi-Region Access Points lauten wie folgt:
  - Pro Konto gibt es maximal 100 Multi-Regions-Zugriffspunkte.
  - Es gibt ein Limit von 17 Regionen für einen einzelnen Multi-Region Access Point.
- Nach der Erstellung eines Multi-Region Access Point können Sie keine Buckets hinzufügen, ändern oder aus der Konfiguration des Multi-Region Access Point entfernen. Um die Buckets zu ändern, müssen Sie den gesamten Multi-Regions-Zugriffspunkt löschen und einen neuen erstellen. Wenn ein kontoübergreifender Bucket in Ihrem Multi-Region Access Point gelöscht wird, besteht die einzige Möglichkeit, diesen Bucket erneut zu verbinden, darin, den Bucket neu zu erstellen und dabei denselben Namen und dieselbe Region in diesem Konto zu verwenden.
- Zugrunde liegende Buckets (in demselben Konto), die in einem Multi-Region Access Point verwendet werden, können erst nach dem Löschen eines Multi-Region Access Point gelöscht werden.
- Alle Anforderungen auf Steuerebene zum Erstellen oder Verwalten von Multi-Region Access Points müssen an die Region `US West (Oregon)` weitergeleitet werden. Für Anforderungen von Multi-Region Access Point auf Steuerebene müssen keine Regionen angegeben werden.
- Für die Failover-Steuerebene von Multi-Region Access Points muss die Anforderung an eine dieser fünf unterstützten Regionen weitergeleitet werden:
  - `US East (N. Virginia)`
  - `US West (Oregon)`
  - `Asia Pacific (Sydney)`
  - `Asia Pacific (Tokyo)`
  - `Europe (Ireland)`
- Ihr Multi-Region Access Point unterstützt nur Buckets in den folgenden AWS-Regionen:
  - `US East (N. Virginia)`
  - `US East (Ohio)`
  - `US West (N. California)`
  - `US West (Oregon)`
  - `Asia Pacific (Mumbai)`
  - `Asia Pacific (Osaka)`
  - `Asia Pacific (Seoul)`

- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- South America (São Paulo)

Weiterleitung der Multi-Regions-Zugriffspunktanforderung

Wenn Sie eine Anforderung über einen Multi-Region Access Point stellen, bestimmt Amazon S3, welche der Buckets, die dem Multi-Region Access Point zugeordnet sind, sich in Ihrer nächster Nähe befinden. Amazon S3 leitet die Anforderung dann an diesen Bucket weiter, unabhängig von der AWS-Region, in der sie sich befindet.

Nachdem der Multi-Region Access Point die Anforderung an den nächstgelegenen Bucket weitergeleitet hat, verarbeitet Amazon S3 die Anforderung so, als ob Sie sie direkt an diesen Bucket gestellt hätten. Multi-Region Access Points kennen den Dateninhalt eines Amazon-S3-Buckets nicht. Daher kann es sein, dass der Bucket, der die Anforderung erhält, die angeforderten Daten nicht enthält. Um konsistente Datensätze in den Amazon-S3-Buckets zu schaffen, die einem Multi-Region Access Point zugeordnet sind, können Sie S3 Cross-Region Replication (CRR, regionsübergreifende Replikation) konfigurieren. Dann kann jeder Bucket die -Anforderung erfolgreich erfüllen.

Amazon S3 leitet Multi-Regions-Zugriffspunktanforderungen gemäß den folgenden Regeln weiter:

- Amazon S3 optimiert Anforderungen, um der Nähe entsprechend erfüllt zu werden. Es untersucht die Buckets, die vom Multi-Region Access Point unterstützt werden, und leitet die Anforderung an den Bucket weiter, der sich in nächster Nähe befindet.
- Wenn die Anforderung eine vorhandene Ressource angibt (z. B. `GetObject`), berücksichtigt Amazon S3 nicht den Namen des Objekts bei der Erfüllung der Anforderung. Selbst wenn ein Objekt in einem Bucket in dem Multi-Region Access Point vorhanden ist, kann Ihre Anforderung

somit an einen Bucket weitergeleitet werden, der das Objekt nicht enthält. Dies führt dazu, dass dem Kunden eine 404-Fehlermeldung zurückgegeben wird.

Um 404-Fehler zu vermeiden, empfehlen wir Ihnen, S3 Cross-Region Replication (CRR) für Ihre Buckets zu konfigurieren. Die Replikation hilft bei der Lösung des potenziellen Problems, das entsteht, wenn sich das gewünschte Objekt in einem Bucket in dem Multi-Region Access Point befindet, nicht jedoch in dem bestimmten Bucket, an den Ihre Anforderung weitergeleitet wurde. Weitere Informationen zum Erstellen einer grundlegenden Replikationskonfiguration finden Sie unter Konfigurieren einer Replikation zur Verwendung mit Multi-Region Access Points.

Um sicherzustellen, dass Ihre Anforderungen mithilfe der gewünschten Objekte erfüllt werden, wird außerdem empfohlen, die Bucket-Versionsverwaltung zu aktivieren und Versions-IDs in Ihre Anforderungen aufzunehmen. Auf diese Weise stellen Sie sicher, dass Sie über die richtige Version des gesuchten Objekts verfügen. Buckets mit aktivierter Versionsverwaltung können auch bei der Wiederherstellung von Objekten nach einem versehentlichen Überschreiben hilfreich sein. Weitere Informationen finden Sie unter Verwenden des S3-Versioning in S3-Buckets.

- Wenn sich die Anforderung auf das Erstellen einer Ressource bezieht (z. B. `PutObject` oder `CreateMultipartUpload`), erfüllt Amazon S3 die Anforderung unter Verwendung des nächstgelegenen Buckets. Betrachten wir beispielsweise ein Videounternehmen, das Video-Uploads von überall auf der Welt unterstützen möchte. Wenn ein Benutzer eine PUT-Anforderung an den Multi-Region Access Point stellt, wird das Objekt in den nächstgelegenen Bucket gestellt. Um das hochgeladene Video dann anderen auf der ganzen Welt mit der geringsten Latenz zum Download zur Verfügung zu stellen, können Sie CRR mit bidirektionaler Replikation verwenden. Wenn Sie CRR mit bidirektionaler Replikation verwenden, bleiben die Inhalte aller Buckets, die dem Multi-Region Access Point zugeordnet sind, synchronisiert. Weitere Informationen zur Verwendung der Replikation mit Multi-Region Access Points finden Sie unter Konfigurieren einer Replikation zur Verwendung mit Multi-Region Access Points.

Failover-Kontrollen für Amazon S3 Multi-Region Access Points

Mit den Failover-Kontrollen für Amazon S3 Multi-Region Access Points können Sie die Geschäftskontinuität bei regionalen Verkehrsunterbrechungen aufrechterhalten und Ihren Anwendungen gleichzeitig eine regionsübergreifende Architektur zur Erfüllung von Compliance- und Redundanzanforderungen bieten. Wenn Ihr regionaler Datenverkehr unterbrochen wird, können Sie mithilfe der Failover-Kontrollen für Multi-Region Access Points auswählen, welche AWS-Regionen hinter einem Amazon S3 Multi-Region Access Point Datenzugriffs- und Speicheranforderungen verarbeiten.

Zur Failover-Unterstützung können Sie Ihren Multi-Region Access Point in einer Aktiv-Passiv-Konfiguration einrichten, wobei der Datenverkehr unter normalen Bedingungen in die aktive Region fließt und sich eine passive Region für den Failover im Standby-Modus befindet.

Wenn Sie beispielsweise ein Failover auf eine AWS-Region Ihrer Wahl durchführen möchten, verlagern Sie den Datenverkehr von Ihrer primären (aktiven) Region in Ihre sekundäre (passive) Region. In einer solchen Aktiv-Passiv-Konfiguration ist ein Bucket aktiv und akzeptiert Datenverkehr, während der andere Bucket passiv ist und keinen Traffic akzeptiert. Der passive Bucket wird für die Notfallwiederherstellung verwendet. Wenn Sie den Failover einleiten, wird der gesamte Datenverkehr (wie `GET`- und `PUT`-Anforderungen) an den Bucket im aktiven Zustand (in einer Region) und weg vom Bucket im passiven Zustand (in einer anderen Region) geleitet.

Wenn Sie die regionsübergreifende Replikation in S3 (CRR) mit bidirektionalen Replikationsregeln aktiviert haben, können Sie Ihre Buckets während eines Failovers synchronisieren. Wenn Sie CRR in einer Aktiv-Aktiv-Konfiguration aktiviert haben, können Amazon S3 Multi-Region Access Points außerdem Daten von dem Bucket-Standort in nächster Nähe abrufen, was die Anwendungsleistung verbessert.

AWS-Region-Support

Mit den Failover-Kontrollen für Amazon S3 Multi-Region Access Points können sich Ihre S3-Buckets in jeder der 17 Regionen befinden, in denen Multi-Region Access Points unterstützt werden. Sie können ein Failover für zwei beliebige Regionen gleichzeitig initiieren.

> **Note**
>
> Obwohl ein Failover nur zwischen zwei Regionen gleichzeitig initiiert wird, können Sie den Routing-Status für mehrere Regionen gleichzeitig in Ihrem Multi-Region Access Point separat aktualisieren.

In den folgenden Themen wird die Verwendung und Verwaltung von Failover-Kontrollen für Amazon S3 Multi-Region Access Points veranschaulicht.

Themen

- Weiterleitungsstatus von Amazon S3 Multi-Region Access Points
- Verwenden von Failover-Kontrollen für Amazon S3 Multi-Region Access Points
- Failover-Kontrollfehler für Amazon S3 Multi-Region Access Points

Weiterleitungsstatus von Amazon S3 Multi-Region Access Points

Ihre Failover-Konfiguration für Amazon S3 Multi-Region Access Points bestimmt den Weiterleitungsstatus der AWS-Regionen, die mit dem Multi-Region Access Point verwendet werden. Sie können Ihren Amazon S3 Multi-Region Access Point so konfigurieren, dass er sich in einem Aktiv-Aktiv- oder Aktiv-Passiv-Zustand befindet.

- Aktiv-Aktiv – In einer Aktiv-Aktiv-Konfiguration werden alle Anfragen automatisch an die nächstgelegenen AWS-Region in Ihrem Multi-Region Access Point gesendet. Nachdem der Multi-Region Access Point so konfiguriert wurde, dass er sich in einem Aktiv-Aktiv-Zustand befindet, können alle Regionen Datenverkehr empfangen. Wenn in einer Aktiv-Aktiv-Konfiguration eine Verkehrsunterbrechung auftritt, wird der Netzwerkverkehr automatisch in eine der aktiven Regionen umgeleitet.
- Aktiv-Passiv – In einer Aktiv-Passiv-Konfiguration empfangen die aktiven Regionen in Ihrem Multi-Region Access Point Traffic und die passiven Regionen nicht. Wenn Sie beabsichtigen, S3-Failover-Kontrollen zu verwenden, um in einer Notfallsituation ein Failover einzuleiten, richten Sie Ihre Multi-Region Access Points in einer Aktiv-Passiv-Konfiguration ein, während Sie die Notfallwiederherstellungsplanung testen und durchführen.

Verwenden von Failover-Kontrollen für Amazon S3 Multi-Region Access Points

In diesem Abschnitt wird erläutert, wie Sie Ihre Amazon S3 Multi-Region Access Points mithilfe der AWS Management Console verwalten und verwenden.

Auf der Detailseite Ihres Multi-Region Access Points finden Sie im Abschnitt Failover configuration (Failover-Konfiguration) in der AWS Management Console zwei Failover-Kontrollen: Edit routing status (Weiterleitungsstatus bearbeiten) und Failover. Verwenden Sie diese Steuerelemente wie folgt:

- Edit routing status (Weiterleitungsstatus bearbeiten) – Sie können den Weiterleitungsstatus von bis zu 17 AWS-Regionen in einer einzigen Anforderung für Ihren Multi-Region Access Point manuell bearbeiten, indem Sie Edit routing status (Weiterleitungsstatus bearbeiten) auswählen. Sie können die Option Edit routing status (Weiterleitungsstatus bearbeiten) für folgende Zwecke verwenden:
  - Zum Festlegen oder Bearbeiten der Weiterleitungsstatus einer oder mehrerer Regionen in Ihrem Multi-Region Access Point
  - Zum Erstellen einer Failover-Konfiguration für Ihren Multi-Region Access Point, indem Sie zwei Regionen so konfigurieren, dass sie sich in einem Aktiv-Passiv-Zustand befinden
  - So führen Sie ein manuelles Failover für Ihre Regionen durch

  - So leiten Sie Traffic manuell zwischen Regionen um
- Failover – Wenn Sie ein Failover initiieren, indem Sie Failover auswählen, aktualisieren Sie nur den Weiterleitungsstatus von zwei Regionen, die bereits so konfiguriert sind, dass sie sich in einem Aktiv-Passiv-Zustand befinden. Während eines Failovers, das Sie initiiert haben, indem Sie Failover auswählen, wird der Weiterleitungsstatus zwischen den beiden Regionen automatisch gewechselt.

Bearbeiten des Weiterleitungsstatus der Regionen in Ihrem Multi-Region Access Point

Sie können den Weiterleitungsstatus von bis zu 17 AWS-Regionen in einer einzigen Anforderung für Ihren Multi-Region Access Point manuell aktualisieren, indem Sie auf der Detailseite Ihres Multi-Region Access Points im Abschnitt Failover configuration (Failover-Konfiguration) die Option Edit routing status (Weiterleitungsstatus bearbeiten) auswählen. Wenn Sie jedoch ein Failover initiieren, indem Sie Failover auswählen, aktualisieren Sie nur den Weiterleitungsstatus von zwei Regionen, die bereits so konfiguriert sind, dass sie sich in einem Aktiv-Passiv-Zustand befinden. Während eines Failovers, das Sie initiiert haben, indem Sie Failover auswählen, wird der Weiterleitungsstatus zwischen den beiden Regionen automatisch gewechselt.

Sie können die Option Edit routing status (Weiterleitungsstatus bearbeiten) (wie im folgenden Verfahren beschrieben) für die folgenden Zwecke verwenden:

- Zum Festlegen oder Bearbeiten der Weiterleitungsstatus einer oder mehrerer Regionen in Ihrem Multi-Region Access Point
- Zum Erstellen einer Failover-Konfiguration für Ihren Multi-Region Access Point, indem Sie zwei Regionen so konfigurieren, dass sie sich in einem Aktiv-Passiv-Zustand befinden
- So führen Sie ein manuelles Failover für Ihre Regionen durch
- So leiten Sie Traffic manuell zwischen Regionen um

Verwenden der S3-Konsole

So aktualisieren Sie den Weiterleitungsstatus der Regionen in Ihrem Multi-Region Access Point

1. Melden Sie sich bei der AWS-Managementkonsole an.
2. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
3. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
4. Wählen Sie den Multi-Region Access Point aus, den Sie aktualisieren möchten.

5. Wählen Sie die Registerkarte Replication and failover (Replikation und Failover) aus.
6. Wählen Sie eine oder mehrere Regionen aus, deren Weiterleitungsstatus Sie bearbeiten möchten.

> **Note**
>
> Damit ein Failover initiiert wird, muss mindestens eine AWS-Region in Ihrem Multi-Region Access Point als Active (Aktiv) und eine Region als Passive (Passiv) festgelegt sein.

7. Wählen Sie Edit routing status (Weiterleitungsstatus bearbeiten) aus.
8. Wählen Sie in dem daraufhin angezeigten Dialogfeld Active (Aktiv) oder Passive (Passiv) als Routing status (Weiterleitungsstatus) für jede Region aus.

   Ein aktiver Status ermöglicht die Weiterleitung von Verkehr an die Region. Ein passiver Status verhindert, dass jeglicher Verkehr an die Region weitergeleitet wird.

   Wenn Sie eine Failover-Konfiguration für Ihren Multi-Region Access Point erstellen oder ein Failover initiieren, muss mindestens eine AWS-Region in Ihrem Multi-Region Access Point als Active (Aktiv) und eine Region als Passive (Passiv) festgelegt sein.
9. Wählen Sie Save routing status (Weiterleitungsstatus speichern) aus. Es dauert etwa 2 Minuten, bis der Verkehr umgeleitet wird.

Nachdem Sie den Weiterleitungssatus der AWS-Regionen für Ihren Multi-Region Access Point übermittelt haben, können Sie die Änderungen des Weiterleitungsstatus überprüfen. Zur Überprüfung dieser Änderungen rufen Sie Amazon CloudWatch unter https://console.aws.amazon.com/cloudwatch auf. Hier können Sie die Verlagerung Ihres Datenanforderungsverkehrs von Amazon S3 (z. B. `GET`- und `PUT`-Anforderungen) zwischen aktiven und passiven Regionen überwachen. Bestehende Verbindungen werden während des Failovers nicht beendet. Bestehende Verbindungen werden fortgesetzt, bis der Erfolgs- bzw. Misserfolgsstatus angezeigt wird.

Verwendung von AWS CLI

> **Note**
>
> Sie können AWS CLI-Weiterleitungsbefehle für den Multi-Region Access Point in einer dieser fünf Regionen ausführen:

- ap-southeast-2
- ap-northeast-1
- us-east-1
- us-west-2
- eu-west-1

Mit dem folgenden Beispielbefehl wird die aktuelle Weiterleitungskonfiguration Ihres Multi-Region Access Points aktualisiert. Wenn Sie den aktiven oder passiven Status eines Buckets aktualisieren möchten, legen Sie den Wert `TrafficDialPercentage` auf `100` für aktiv und auf `0` für passiv fest. In diesem Beispiel ist *DOC-EXAMPLE-BUCKET-1* auf aktiv und *DOC-EXAMPLE-BUCKET-2* auf passiv eingestellt. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control submit-multi-region-access-point-routes
--region ap-southeast-2
--account-id 111122223333
--mrap MultiRegionAccessPoint_ARN
--route-updates Bucket=DOC-EXAMPLE-BUCKET-1,TrafficDialPercentage=100
                Bucket=DOC-EXAMPLE-BUCKET-2,TrafficDialPercentage=0
```

Mit dem folgenden Beispielbefehl wird die aktualisierte Weiterleitungskonfiguration Ihres Multi-Region Access Points abgerufen. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control get-multi-region-access-point-routes
--region eu-west-1
--account-id 111122223333
--mrap MultiRegionAccessPoint_ARN
```

Initiieren eines Failovers

Wenn Sie einen Failover einleiten, indem Sie auf der Detailseite Ihres Multi-Region Access Points im Abschnitt Failover configuration (Failover-Konfiguration) die Option Failover auswählen, wird der Amazon-S3-Anforderungsverkehr automatisch auf eine alternative AWS-Region umgeleitet. Der Failover-Prozess ist innerhalb von 2 Minuten abgeschlossen.

Sie können ein Failover für zwei beliebige AWS-Regionen Regionen gleichzeitig initiieren (zwei der 17 Regionen, in denen Multi-Region Access Points unterstützt werden). Failover-Ereignisse werden dann in AWS CloudTrail protokolliert. Nach Abschluss des Failovers können Sie den Amazon-S3-Traffic und alle Aktualisierungen der Verkehrweiterleitungen auf die neue aktive Region in Amazon CloudWatch überwachen.

> **Important**
>
> Damit alle Metadaten und Objekte während der Datenreplikation bucketübergreifend synchronisiert werden, empfehlen wir, bidirektionale Replikationsregeln zu erstellen und die Synchronisierung von Replikatänderungen zu aktivieren, bevor Sie Ihre Failover-Kontrollen konfigurieren.
> Bidirektionale Replikationsregeln tragen dazu bei, sicherzustellen, dass wenn Daten in den Amazon-S3-Bucket geschrieben werden, auf den der Datenverkehr bei einem Failover zurückgreift, diese Daten dann zurück in den Quell-Bucket repliziert werden. Durch die Synchronisierung von Replikatänderungen wird sichergestellt, dass die Objektmetadaten während der bidirektionalen Replikation auch zwischen Buckets synchronisiert werden. Weitere Informationen zum Konfigurieren der Replikation zur Unterstützung von Failover finden Sie unter the section called “Bucket-Replikation”.

So leiten Sie ein Failover zwischen replizierten Buckets ein

1. Melden Sie sich bei der AWS-Managementkonsole an.
2. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
3. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
4. Wählen Sie den Multi-Region Access Point aus, den Sie zum Initiieren des Failovers verwenden möchten.
5. Wählen Sie die Registerkarte Replication and failover (Replikation und Failover) aus.
6. Scrollen Sie nach unten zum Abschnitt Failover configuration (Failover-Konfiguration) und wählen Sie zwei AWS-Regionen aus.

> **Note**
>
> Damit ein Failover initiiert wird, muss mindestens eine AWS-Region in Ihrem Multi-Region Access Point als Active (Aktiv) und eine Region als Passive (Passiv) festgelegt

sein. Ein aktiver Status ermöglicht die Weiterleitung von Verkehr an eine Region. Ein passiver Status verhindert, dass jeglicher Verkehr an die Region weitergeleitet wird.

7. Klicken Sie auf Failover.
8. Wählen Sie im Dialogfeld erneut Failover aus, um den Failover-Vorgang einzuleiten. Während dieses Vorgangs werden die Weiterleitungsstatus der beiden Regionen automatisch gewechselt. Der gesamte neue Traffic wird an die Region geleitet, die aktiv wird, und der Verkehr wird nicht mehr an die Region weitergeleitet, die passiv wird. Es dauert etwa 2 Minuten, bis der Verkehr umgeleitet wird.

   Nachdem Sie den Failover-Prozess eingeleitet haben, können Sie Ihre Verkehrsänderungen überprüfen. Zur Überprüfung dieser Änderungen rufen Sie Amazon CloudWatch unter https://console.aws.amazon.com/cloudwatch auf. Hier können Sie die Verlagerung Ihres Datenanforderungsverkehrs von Amazon S3 (z. B. `GET`- und `PUT`-Anforderungen) zwischen aktiven und passiven Regionen überwachen. Bestehende Verbindungen werden während des Failovers nicht beendet. Bestehende Verbindungen werden fortgesetzt, bis der Erfolgs- bzw. Misserfolgsstatus angezeigt wird.

Anzeigen Ihrer Weiterleitungskontrollen für Amazon S3 Multi-Region Access Points

Verwenden der S3-Konsole

So zeigen Sie Ihre Weiterleitungskontrollen für Ihren Amazon S3 Multi-Region Access Point an

1. Melden Sie sich bei der AWS-Managementkonsole an.
2. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
3. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
4. Wählen Sie den Multi-Region Access Point aus, den Sie überprüfen möchten.
5. Wählen Sie die Registerkarte Replication and failover (Replikation und Failover) aus. Auf dieser Seite werden die Konfigurationsdetails für die Weiterleitung und eine Zusammenfassung für Ihren Multi-Region Access Point, die zugehörigen Replikationsregeln und Replikationsmetriken angezeigt. Sie können den Weiterleitungsstatus Ihrer Regionen im Abschnitt Failover configuration (Failover-Konfiguration) sehen.

Verwendung von AWS CLI

Mit dem folgenden AWS CLI-Beispielbefehl wird die aktuelle Weiterleitungskonfiguration Ihres Multi-Region Access Points für die angegebene Region abgerufen. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control get-multi-region-access-point-routes
--region eu-west-1
--account-id 111122223333
--mrap MultiRegionAccessPoint_ARN
```

> **Note**
>
> Dieser Befehl kann nur für diese fünf Regionen ausgeführt werden:
>
> - ap-southeast-2
> - ap-northeast-1
> - us-east-1
> - us-west-2
> - eu-west-1

Failover-Kontrollfehler für Amazon S3 Multi-Region Access Points

Wenn Sie die Failover-Konfiguration für Ihren Multiregion Access Point aktualisieren, tritt möglicherweise einer der folgenden Fehler auf:

- HTTP 400 Ungültige Anforderung: Dieser Fehler kann auftreten, wenn Sie bei der Aktualisierung Ihrer Failover-Konfiguration einen ungültigen ARN für den Multi-Region Access Point eingeben. Sie können den ARN Ihres Multi-Region Access Points anhand der Richtlinie des Multi-Region Access Points überprüfen. Informationen zur Überprüfung oder Aktualisierung Ihrer der Richtlinie Ihres Multi-Region Access Points finden Sie unter Bearbeiten der Richtlinie für den Multi-Region Access Point. Dieser Fehler kann auch auftreten, wenn Sie beim Aktualisieren der Failover-Kontrollen für Ihren Amazon S3 Multi-Region Access Point eine leere oder zufällige Zeichenfolge verwenden. Für ARNs von Multi-Region Access Points muss folgendes Format verwendet werden:

  `arn:aws:s3::account-id:accesspoint/MultiRegionAccessPoint_alias`

- HTTP 503 Slow Down: Dieser Fehler tritt auf, wenn Sie innerhalb kurzer Zeit zu viele Anforderungen senden. Abgelehnte Anforderungen führen zu einem Fehler.
- HTTP 409 Conflict: Dieser Fehler tritt auf, wenn zwei oder mehr gleichzeitige Aktualisierungsanforderungen für die Weiterleitungskonfiguration an einen einzelnen Multi-Region Access Point gerichtet sind. Die erste Anforderung ist erfolgreich, aber alle anderen Anforderungen schlagen mit einem Fehler fehl.
- HTTP 405 Method Not Allowed: Dieser Fehler tritt auf, wenn Sie beim Initiieren des Failovers einen Multi-Region Access Point mit nur einer AWS-Region ausgewählt haben. Sie müssen zwei Regionen auswählen, bevor Sie den Failover einleiten können. Andernfalls wird ein Fehler zurückgegeben.

Konfigurieren einer Replikation zur Verwendung mit Multi-Region Access Points

Wenn Sie eine Anforderung an einen Endpunkt eines Multi-Region Access Point stellen, leitet Amazon S3 die Anforderung automatisch an den Ihnen nächstgelegenen Bucket weiter. Amazon S3 berücksichtigt die Inhalte der Anforderung bei dieser Entscheidung nicht. Wenn Sie eine Anforderung stellen, um ein Objekt mit `GET` abzurufen, wird Ihre Anforderung möglicherweise an einen Bucket weitergeleitet, der keine Kopie dieses Objekts besitzt. In diesem Fall erhalten Sie den HTTP-Statuscode 404 (Nicht gefunden). Weitere Informationen zur Weiterleitung von Multi-Region-Access-Point-Anforderungen finden Sie unter the section called “Weiterleitung von Anforderungen”.

Wenn Sie möchten, dass der Multi-Region Access Point das Objekt unabhängig davon wiederherstellen kann, welcher Bucket die Anforderung empfängt, müssen Sie die regionsübergreifende Replikation von Amazon S3 (CRR) konfigurieren.

Betrachten Sie einen Multi-Region Access Point mit drei Buckets:

- Ein Bucket mit dem Namen `my-bucket-usw2` in der Region `us-west-2`, der das Objekt `my-image.jpg` enthält
- Ein Bucket mit dem Namen `my-bucket-aps1` in der Region `ap-south-1`, der das Objekt `my-image.jpg` enthält
- Ein Bucket mit dem Namen `my-bucket-euc1` in der Region `eu-central-1`, der das Objekt `my-image.jpg` nicht enthält

Wenn Sie in dieser Situation eine `GetObject`-Anforderung für das Objekt `my-image.jpg` stellen, hängt der Erfolg dieser Anforderung davon ab, welcher Bucket Ihre Anforderung empfängt. Da Amazon S3 den Inhalt der Anforderung nicht berücksichtigt, kann es Ihre `GetObject`-Anforderung an den `my-bucket-euc1`-Bucket weiterleiten, wenn dieser Bucket aus nächster Nähe antwortet. Obwohl sich Ihr Objekt in einem Bucket im Multi-Region Access Point befindet, erhalten Sie einen HTTP-Fehler 404 Not Found, da der einzelne Bucket, der Ihre Anforderung empfangen hat, das Objekt nicht enthielt.

Durch das Aktivieren der regionsübergreifenden Replikationsfunktion (CRR) kann dieses Ergebnis vermieden werden. Mit den entsprechenden Replikationsregeln wird das `my-image.jpg`-Objekt in den `my-bucket-euc1`-Bucket kopiert. Wenn Amazon S3 Ihre Anforderung an diesen Bucket weiterleitet, können Sie das Objekt daher jetzt abrufen.

Die Replikation funktioniert normal mit Buckets, die einem Multi-Regions-Zugriffspunkt zugewiesen sind. Amazon S3 führt keine spezielle Replikationsbehandlung mit Buckets aus, die sich in Multi-Region Access Points befinden. Weitere Informationen zum Konfigurieren einer Replikation in Ihren Buckets finden Sie unter Einrichten der Replikation.

Empfehlungen zur Verwendung der Replikationsfunktion mit Multi-Region Access Points

Für beste Replikationsleistung bei der Arbeit mit Multi-Region Access Points empfehlen wir Folgendes:

- Konfigurieren Sie die Begrenzung der S3-Replikationszeit (S3 RTC). Um Ihre Daten innerhalb eines vorhersehbaren Zeitraums über verschiedene Regionen hinweg zu replizieren, können Sie S3 RTC verwenden. S3 RTC repliziert 99,99 Prozent der neuen in Amazon S3 gespeicherten Objekte innerhalb von 15 Minuten (gestützt auf ein Service Level Agreement). Weitere Informationen finden Sie unter the section called “Verwenden der S3-Replikationszeitkontrolle”. Für S3 RTC fallen zusätzliche Gebühren an. Informationen finden Sie unter Amazon S3 – Preise.
- Verwenden Sie die bidirektionale Replikation, um die Synchronisierung von Buckets zu unterstützen, wenn Buckets über den Multi-Region Access Point aktualisiert werden. Weitere Informationen finden Sie unter the section called “Erstellen von bidirektionalen Replikationsregeln für Ihren Multi-Region Access Point”.
- Erstellen Sie kontoübergreifende Multi-Region Access Points für die Replikation von Daten in Buckets in separaten AWS-Konten. Dieses Vorgehen ermöglicht eine Trennung auf Kontoebene, sodass auf Daten von verschiedenen Konten in anderen Regionen als dem Quell-Bucket zugegriffen werden kann und diese repliziert werden können. Die Einrichtung von kontoübergreifenden Multi-Region Access Points ist ohne zusätzliche Kosten möglich. Wenn Sie

Bucket-Eigentümer, jedoch nicht Eigentümer des Multi-Region Access Point sind, zahlen Sie nur für die Datenübertragung und die Anforderungskosten. Eigentümer von Multi-Region Access Points tragen die Kosten für die Datenweiterleitung und die Internetbeschleunigung. Weitere Informationen finden Sie unter Amazon S3 – Preise.

- Aktivieren Sie die Synchronisierung von Replikatänderungen für jede Replikationsregel, um auch Metadatenänderungen an Ihren Objekten zu synchronisieren. Weitere Informationen finden Sie unter Aktivieren der Synchronisierung von Replikatänderungen.
- Aktivieren Sie Amazon-CloudWatch-Metriken, um Replikationsereignisse zu überwachen. Es fallen Gebühren für CloudWatch-Metriken an. Weitere Informationen hierzu finden Sie unter Amazon CloudWatch – Preise.

Themen

Erstellen von unidirektionalen Replikationsregeln für Ihren Multi-Region Access Point

Replikationsregeln ermöglichen ein automatisches und asynchrones Kopieren von Objekten über Buckets hinweg. Eine unidirektionale Replikationsregel hilft sicherzustellen, dass Daten vollständig aus einem Quell-Bucket in einer AWS-Region in einen Ziel-Bucket in einer anderen Region repliziert werden. Wenn unidirektionale Replikation eingerichtet ist, wird eine Replikationsregel vom Quell-Bucket (DOC-EXAMPLE-BUCKET-1) zum Ziel-Bucket (DOC-EXAMPLE-BUCKET-2) erstellt. Wie alle Replikationsregeln können Sie die unidirektionale Replikationsregel entweder auf den gesamten Amazon-S3-Bucket oder auf eine anhand eines Präfixes oder von Objekt-Tags gefilterte Teilmenge von Objekten anwenden.

Important

Wir empfehlen eine unidirektionale Replikation, wenn Ihre Benutzer die Objekte in Ihren Ziel-Buckets nur verwenden werden. Wenn Ihre Benutzer die Objekte in Ihren Ziel-Buckets hochladen oder ändern, verwenden Sie die bidirektionale Replikation, damit alle Ihre Buckets synchronisiert bleiben. Wir empfehlen eine bidirektionale Replikation auch, wenn Sie Ihren Multi-Region Access Point für Failover verwenden möchten. Informationen zum Einrichten der

bidirektionalen Replikation finden Sie unter the section called “Erstellen von bidirektionalen Replikationsregeln für Ihren Multi-Region Access Point”.

So erstellen Sie eine unidirektionale Replikationsregel für Ihren Multi-Region Access Point

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie den Namen Ihres Multi-Region Access Point aus.
4. Wählen Sie die Registerkarte Replication and failover (Replikation und Failover) aus.
5. Scrollen Sie nach unten zum Abschnitt Replication rules (Replikationsregeln) und wählen Sie dann Create replication rules (Replikationsregeln erstellen) aus. Vergewissern Sie sich, dass Sie über ausreichende Berechtigungen verfügen, um die Replikationsregel zu erstellen. Andernfalls wird die Versionsverwaltung deaktiviert.

   > **Note**
   >
   > Sie können nur für Buckets in Ihrem eigenen Konto Replikationsregeln erstellen. Replikationsregeln für externe Buckets müssen die Bucket-Eigentümer selbst erstellen.

6. Wählen Sie auf der Seite Replikationsregeln erstellen die Vorlage zum Replizieren von Objekten aus einem oder mehreren Quell-Buckets in einen oder mehrere Ziel-Buckets aus.

   > **Important**
   >
   > Wenn Sie Replikationsregeln mithilfe dieser Vorlage erstellen, ersetzen diese alle vorhandenen Replikationsregeln, die dem Bucket bereits zugewiesen sind.
   > Wenn Sie Replikationsregeln hinzufügen oder bestehende Regeln ändern möchten, anstatt sie zu ersetzen, wechseln Sie in der Konsole zur Registerkarte Management (Verwaltung) der einzelnen Buckets und bearbeiten Sie dann die Regeln im Abschnitt Replication rules (Replikationsregeln). Sie können bestehende Replikationsregeln auch mithilfe der AWS CLI, SDKs oder der REST-API hinzufügen oder ändern. Weitere Informationen finden Sie unter Replikations-Konfiguration.

7. Wählen Sie im Abschnitt Quelle und Ziel unter Quell-Buckets einen oder mehrere Buckets aus, aus dem/denen Sie Objekte replizieren möchten. Für alle für die Replikation ausgewählten

Buckets (Quell- und Ziel-Buckets) muss die S3-Versionsverwaltung aktiviert sein und jeder Bucket muss sich in einer anderen AWS-Region befinden. Weitere Informationen zu S3 Versioning finden Sie unter Verwenden der Versionsverwaltung in Amazon-S3-Buckets.

Wählen Sie unter Ziel-Buckets einen oder mehrere Buckets aus, in die Sie Objekte replizieren möchten.

> **Note**
>
> Stellen Sie sicher, dass Sie über die erforderlichen Lese- und Replikationsberechtigungen verfügen, um die Replikation einzurichten. Andernfalls treten Fehler auf. Weitere Informationen finden Sie unter Erstellen einer IAM-Rolle.

8. Wählen Sie im Abschnitt Replication rule configuration (Konfiguration der Replikationsregel) aus, ob die Replikationsregel bei ihrer Erstellung Enabled (Aktiviert) oder Disabled (Deaktiviert) sein soll.

> **Note**
>
> Sie können keinen Namen in das Feld Replication rule name (Name der Replikationsregel) eingeben. Die Namen der Replikationsregeln werden basierend auf Ihrer Konfiguration generiert, wenn Sie die Replikationsregel erstellen.

9. Wählen Sie im Abschnitt Scope (Bereich) den entsprechenden Bereich für Ihre Replikation aus.

   - Um den gesamten Bucket zu replizieren, wählen Sie Apply to all objects in the bucket (Auf alle Objekte im Bucket anwenden).
   - Um eine Teilmenge der Objekte im Bucket zu replizieren, wählen Sie Limit the scope of this rule using one or more filters (Geltungsbereich dieser Regel mit einem oder mehreren Filtern einschränken) aus.

     Sie können Ihre Objekte nach Präfix, Objekt-Tags oder einer Kombination aus beiden filtern.

     - Wenn Sie die Replikation auf alle Objekte beschränken möchten, die mit derselben Zeichenfolge beginnen (z. B. `pictures`), geben Sie ein Präfix in das Feld Prefix (Präfix) ein.

       Wenn Sie ein Präfix eingeben, bei dem es sich um den Namen eines Ordners handelt, müssen Sie ein Trennzeichen wie z. B. `/` (Schrägstrich) verwenden, um die Hierarchieebene

des Ordners anzugeben (z. B. `pictures/`). Weitere Informationen zu Präfixen finden Sie unter Organisieren von Objekten mit Präfixen.

- Um alle Objekte mit einem oder mehreren Objekt-Tags zu replizieren, wählen Sie Add tag (Tag hinzufügen) aus und geben Sie das Schlüssel-Wert-Paar in die Felder ein. Wiederholen Sie den Vorgang, um ein weiteres Tag hinzuzufügen. Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

10. Scrollen Sie nach unten zum Abschnitt Additional replication options (Zusätzliche Replikationsoptionen) und wählen Sie die Replikationsoptionen aus, die Sie anwenden möchten.

> **Note**
>
> Wir empfehlen Ihnen, folgende Optionen anzuwenden:
>
> - Replication time control (RTC) Begrenzung der Replikationszeit (RTC) – Um Ihre Daten innerhalb eines vorhersehbaren Zeitraums über verschiedene Regionen hinweg zu replizieren, können Sie die Begrenzung der S3-Replikationszeit (S3 RTC) verwenden. S3 RTC repliziert 99,99 Prozent der neuen in Amazon S3 gespeicherten Objekte innerhalb von 15 Minuten (gestützt auf ein Service Level Agreement). Weitere Informationen finden Sie unter the section called “Verwenden der S3-Replikationszeitkontrolle”.
> - Replication metrics and notifications (Replikationsmetriken und -benachrichtigungen) – Aktivieren Sie Amazon-CloudWatch-Metriken, um Replikationsereignisse zu überwachen.
> - Replikation der Löschmarkierung – Durch S3-Löschvorgänge erstellte Löschmarkierungen werden repliziert. Durch Lebenszyklusregeln erstellte Löschmarkierungen werden nicht repliziert. Weitere Informationen finden Sie unter Replizieren von Löschmarkierungen auf Buckets.
>
> Es fallen zusätzliche Gebühren für S3 RTC und CloudWatch-Replikationsmetriken und -benachrichtigungen an. Weitere Informationen finden Sie unter Amazon S3 – Preise und Amazon CloudWatch – Preise.

11. Wenn Sie eine neue Replikationsregel schreiben, die eine bestehende ersetzt, wählen Sie I acknowledge that by choosing Create replication rules, these existing replication rules will be

overwritten (Ich bestätige, dass diese vorhandenen Replikationsregeln durch Auswählen von „Replikationsregeln erstellen“ überschrieben werden).

12. Wählen Sie Replikationsregeln erstellen aus, um Ihre neue unidirektionale Replikationsregel zu erstellen und zu speichern.

Erstellen von bidirektionalen Replikationsregeln für Ihren Multi-Region Access Point

Replikationsregeln ermöglichen ein automatisches und asynchrones Kopieren von Objekten über Buckets hinweg. Eine bidirektionale Replikationsregel stellt sicher, dass Daten zwischen zwei oder mehr Buckets in verschiedenen AWS-Regionen vollständig synchronisiert werden. Wenn die bidirektionale Replikation eingerichtet ist, wird eine Replikationsregel vom Quell-Bucket (DOC-EXAMPLE-BUCKET-1) zu dem Bucket, der die Replikate enthält (DOC-EXAMPLE-BUCKET-2) erstellt. Anschließend wird eine zweite Replikationsregel von dem Bucket, der die Replikate enthält (DOC-EXAMPLE-BUCKET-2), zum Quell-Bucket (DOC-EXAMPLE-BUCKET-1) erstellt.

Wie alle Replikationsregeln können Sie die bidirektionale Replikationsregel entweder auf den gesamten Amazon-S3-Bucket oder auf eine anhand eines Präfixes oder von Objekt-Tags gefilterte Teilmenge von Objekten anwenden. Sie können auch Metadatenänderungen an Ihren Objekten u synchronisieren, indem Sie die Synchronisierung von Replikatänderungen für jede Replikationsregel aktivieren. Sie können die Synchronisierung von Replikatänderungen über die Amazon-S3-Konsole, die AWS CLI, die AWS-SDKs, die Amazon-S3-REST-API oder AWS CloudFormation aktivieren.

Um den Replikationsfortschritt von Objekten und Objektmetadaten in Amazon CloudWatch zu überwachen, aktivieren Sie S3-Replikationsmetriken und -benachrichtigungen. Weitere Informationen finden Sie unter Überwachen des Fortschritts mit Replikationsmetriken und Amazon-S3-Ereignisbenachrichtigungen.

So erstellen Sie eine bidirektionale Replikationsregel für Ihren Multi-Region Access Point

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie den Namen des Multi-Region Access Point aus, den Sie aktualisieren möchten.
4. Wählen Sie die Registerkarte Replication and failover (Replikation und Failover) aus.
5. Scrollen Sie nach unten zum Abschnitt Replication rules (Replikationsregeln) und wählen Sie dann Create replication rules (Replikationsregeln erstellen) aus.

6. Wählen Sie auf der Seite Create replication rules (Replikationsregeln erstellen) die Vorlage Replicate objects among all specified buckets (Replizieren von Objekten zwischen allen angegebenen Buckets) aus. Die Vorlage Replicate objects among all specified buckets (Replizieren von Objekten zwischen allen angegebenen Buckets) richtet die bidirektionale Replikation (mit Failover-Funktionen) für Ihre Buckets ein.

   > **Important**
   >
   > Wenn Sie Replikationsregeln mithilfe dieser Vorlage erstellen, ersetzen diese alle vorhandenen Replikationsregeln, die dem Bucket bereits zugewiesen sind.
   > Wenn Sie Replikationsregeln hinzufügen oder bestehende Regeln ändern möchten, anstatt sie zu ersetzen, wechseln Sie in der Konsole zur Registerkarte Management (Verwaltung) der einzelnen Buckets und bearbeiten Sie dann die Regeln im Abschnitt Replication rules (Replikationsregeln). Sie können bestehende Replikationsregeln auch über die AWS CLI, AWS-SDKs oder die Amazon-S3-REST-API hinzufügen oder ändern. Weitere Informationen finden Sie unter Replikations-Konfiguration.

7. Wählen Sie im Abschnitt Buckets mindestens zwei Buckets aus, aus denen Sie Objekte replizieren möchten. Für alle für die Replikation ausgewählten Buckets muss S3 Versioning aktiviert sein und jeder Bucket muss sich in einer anderen AWS-Region befinden. Weitere Informationen zu S3 Versioning finden Sie unter Verwenden der Versionsverwaltung in Amazon-S3-Buckets.

   > **Note**
   >
   > Stellen Sie sicher, dass Sie über die erforderlichen Lese- und Replikationsberechtigungen verfügen, um die Replikation einzurichten. Andernfalls treten Fehler auf. Weitere Informationen finden Sie unter Erstellen einer IAM-Rolle.

8. Wählen Sie im Abschnitt Replication rule configuration (Konfiguration der Replikationsregel) aus, ob die Replikationsregel bei ihrer Erstellung Enabled (Aktiviert) oder Disabled (Deaktiviert) sein soll.

> **Note**
>
> Sie können keinen Namen in das Feld Replication rule name (Name der Replikationsregel) eingeben. Die Namen der Replikationsregeln werden basierend auf Ihrer Konfiguration generiert, wenn Sie die Replikationsregel erstellen.

9. Wählen Sie im Abschnitt Scope (Bereich) den entsprechenden Bereich für Ihre Replikation aus.

   - Um den gesamten Bucket zu replizieren, wählen Sie Apply to all objects in the bucket (Auf alle Objekte im Bucket anwenden).
   - Um eine Teilmenge der Objekte im Bucket zu replizieren, wählen Sie Limit the scope of this rule using one or more filters (Geltungsbereich dieser Regel mit einem oder mehreren Filtern einschränken) aus.

     Sie können Ihre Objekte nach Präfix, Objekt-Tags oder einer Kombination aus beiden filtern.

     - Wenn Sie die Replikation auf alle Objekte beschränken möchten, die mit derselben Zeichenfolge beginnen (z. B. `pictures`), geben Sie ein Präfix in das Feld Prefix (Präfix) ein.

       Wenn Sie ein Präfix eingeben, das den Namen eines Ordners darstellt, müssen Sie `/` (Schrägstrich) als letztes Zeichen eingeben (z. B. `pictures/`).
     - Um alle Objekte mit einem oder mehreren Objekt-Tags zu replizieren, wählen Sie Add tag (Tag hinzufügen) aus und geben Sie das Schlüssel-Wert-Paar in die Felder ein. Wiederholen Sie den Vorgang, um ein weiteres Tag hinzuzufügen. Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

10. Scrollen Sie nach unten zum Abschnitt Additional replication options (Zusätzliche Replikationsoptionen) und wählen Sie die Replikationsoptionen aus, die Sie anwenden möchten.

> **Note**
>
> Wir empfehlen Ihnen, die folgenden Optionen anzuwenden, insbesondere wenn Sie Ihren Multi-Region Access Point so konfigurieren möchten, dass er Failover unterstützt:
>
> - Replication time control (RTC) Begrenzung der Replikationszeit (RTC) – Um Ihre Daten innerhalb eines vorhersehbaren Zeitraums über verschiedene Regionen hinweg zu replizieren, können Sie die Begrenzung der S3-Replikationszeit (S3 RTC)

verwenden. S3 RTC repliziert 99,99 Prozent der neuen in Amazon S3 gespeicherten Objekte innerhalb von 15 Minuten (gestützt auf ein Service Level Agreement). Weitere Informationen finden Sie unter the section called “Verwenden der S3-Replikationszeitkontrolle”.

- Replication metrics and notifications (Replikationsmetriken und -benachrichtigungen) – Aktivieren Sie Amazon-CloudWatch-Metriken, um Replikationsereignisse zu überwachen.
- Replikation der Löschmarkierung – Durch S3-Löschvorgänge erstellte Löschmarkierungen werden repliziert. Durch Lebenszyklusregeln erstellte Löschmarkierungen werden nicht repliziert. Weitere Informationen finden Sie unter Replizieren von Löschmarkierungen auf Buckets.
- Replica modification sync (Synchronisierung von Replikatänderungen) – Aktivieren Sie die Synchronisierung von Replikatänderungen für jede Replikationsregel, um auch Metadatenänderungen an Ihren Objekten zu synchronisieren. Weitere Informationen finden Sie unter Aktivieren der Synchronisierung von Replikatänderungen.

Es fallen zusätzliche Gebühren für S3 RTC und CloudWatch-Replikationsmetriken und -benachrichtigungen an. Weitere Informationen finden Sie unter Amazon S3 – Preise und Amazon CloudWatch – Preise.

11. Wenn Sie eine neue Replikationsregel schreiben, die eine bestehende ersetzt, wählen Sie I acknowledge that by choosing Create replication rules, these existing replication rules will be overwritten (Ich bestätige, dass diese vorhandenen Replikationsregeln durch Auswählen von „Replikationsregeln erstellen“ überschrieben werden).
12. Wählen Sie Create replication rules (Replikationsregeln erstellen), um Ihre neuen bidirektionalen Replikationsregeln zu erstellen und zu speichern.

Anzeigen der Replikationsregeln für Ihren Multi-Region Access Point

Mit Multi-Region Access Points können Sie entweder unidirektionale oder bidirektionale Replikationsregeln einrichten. Informationen zur Verwaltung Ihrer Replikationsregeln finden Sie unter Verwalten von Replikationsregeln mit der Amazon-S3-Konsole.

So zeigen Sie die Replikationsregeln für Ihren Multi-Region Access Point an

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Multi-Region Access Points aus.
3. Wählen Sie den Namen Ihres Multi-Region Access Point aus.
4. Wählen Sie die Registerkarte Replication and failover (Replikation und Failover) aus.
5. Scrollen Sie nach unten bis zum Abschnitt Replikationsregeln. In diesem Abschnitt sind alle Replikationsregeln aufgeführt, die für Ihren Multi-Region Access Point erstellt wurden.

> **Note**
>
> Wenn Sie diesem Multi-Region Access Point einen Bucket von einem anderen Konto hinzugefügt haben, benötigen Sie die Berechtigung `s3:GetBucketReplication` vom Bucket-Eigentümer, um die Replikationsregeln für diesen Bucket anzeigen zu können.

Verwenden von Multi-Region Access Points mit unterstützten API-Operationen

Amazon S3 unterstützt verschiedene Vorgänge, mit denen Sie Multi-Regions-Zugriffspunkte verwalten können. Amazon S3 verarbeitet einige dieser Vorgänge synchron und einige asynchron. Wenn Sie einen asynchronen Vorgang aufrufen, autorisiert Amazon S3 den angeforderten Vorgang zunächst synchron. Wenn die Autorisierung erfolgreich ist, gibt Amazon S3 ein Token zurück, mit dem Sie den Fortschritt und die Ergebnisse des angeforderten Vorgangs verfolgen können.

> **Note**
>
> Anforderungen, die über die Amazon-S3-Konsole erfolgen, sind immer synchron. Die Konsole wartet, bis die Anforderung abgeschlossen ist, bevor Sie eine weitere Anforderung senden können.

Sie können den aktuellen Status und die Ergebnisse asynchroner Operationen mithilfe der Konsole anzeigen oder `DescribeMultiRegionAccessPointOperation` in der AWS CLI, AWS SDKs oder REST API verwenden. Amazon S3 stellt ein Tracking-Token in der Antwort auf einen asynchronen Vorgang bereit. Sie schließen dieses Tracking-Token als Argument für `DescribeMultiRegionAccessPointOperation` ein. Wenn Sie das Tracking-Token

einschließen, gibt Amazon S3 den aktuellen Status und die Ergebnisse des angegebenen Vorgangs zurück, einschließlich etwaiger Fehler oder relevanter Ressourceninformationen. Amazon S3 führt `DescribeMultiRegionAccessPointOperation`-Vorgänge synchron durch.

Alle Anforderungen auf Steuerebene zum Erstellen oder Verwalten von Multi-Region Access Points müssen an die Region `US West (Oregon)` weitergeleitet werden. Für Anforderungen von Multi-Region Access Point auf Steuerebene müssen keine Regionen angegeben werden. Für die Failover-Steuerebene von Multi-Region Access Points muss die Anforderung an eine der fünf unterstützten Regionen weitergeleitet werden. Weitere Informationen zu unterstützten Regionen für Multi-Region Access Points finden Sie unter Einschränkungen und Beschränkungen des Multi-Regions-Zugriffspunkts.

Darüber hinaus müssen Sie dem Benutzer, der Rolle oder einer anderen AWS Identity and Access Management (IAM)-Entität, die eine Anforderung zur Verwaltung eines Multi-Region Access Point stellt, die `s3:ListAllMyBuckets` Berechtigung erteilen.

Die folgenden Beispiele veranschaulichen, wie Multi-Region Access Points mit kompatiblen Operationen in Amazon S3 verwendet werden.

Themen

- Kompatibilität von Multi-Region Access Points mit - AWS-Services und AWS -SDKs
- Kompatibilität von Multi-Region Access Points mit S3-Operationen
- Anzeigen Ihrer Weiterleitungskonfiguration für Multi-Region Access Points
- Aktualisieren der zugrunde liegende Bucket-Richtlinie von Amazon S3
- Aktualisieren einer Weiterleitungskonfiguration von Multi-Region Access Points
- Hinzufügen eines Objekts zu einem Bucket in Ihrem Multi-Region Access Point
- Abrufen von Objekten von Ihrem Multi-Region Access Point
- Auflisten von Objekten, die in einem Bucket gespeichert sind, der Ihrem Multi-Region Access Point zugrunde liegt
- Verwenden einer vorsignierte URL mit Multi-Region Access Points
- Verwenden eines Buckets, der mit „Zahlung durch den Anforderer“ konfiguriert ist, mit Multi-Region Access Points

Kompatibilität von Multi-Region Access Points mit - AWS-Services und AWS -SDKs

Um einen Multi-Regions-Zugriffspunkt mit Anwendungen zu verwenden, die einen Amazon S3-Bucket-Namen erfordern, verwenden Sie den Amazon-Ressourcennamen (ARN) des Multi-Regions-Zugriffspunkts, wenn Sie Anforderungen mithilfe eines AWS -SDK stellen. Informationen dazu, welche - AWS SDKs mit Multi-Region Access Points kompatibel sind, finden Sie unter Kompatibilität mit - AWS SDKs.

Kompatibilität von Multi-Region Access Points mit S3-Operationen

Sie können die folgenden API-Operationen von Amazon S3 auf Datenebene verwenden, um Aktionen für Objekte in Buckets auszuführen, die Ihrem Multi-Region Access Point zugeordnet sind. Die folgenden S3-Operationen können ARNs von Multi-Region Access Points akzeptieren:

- AbortMultipartUpload
- CompleteMultipartUpload
- CreateMultipartUpload
- DeleteObject
- DeleteObjectTagging
- GetObject
- GetObjectAcl
- GetObjectLegalHold
- GetObjectRetention
- GetObjectTagging
- HeadObject
- ListMultipartUploads
- ListObjectsV2
- ListParts
- PutObject
- PutObjectAcl
- PutObjectLegalHold
- PutObjectRetention
- PutObjectTagging

- RestoreObject
- UploadPart

> **Note**
>
> Multi-Region Access Points unterstützen die API-Operation CopyObject nicht. Stattdessen müssen Sie `CopyObject`-Aktionen direkt zwischen Buckets durchführen.

Sie können die folgenden Amazon-S3-Operationen auf Steuerebene verwenden, um Ihre Multi-Region Access Points zu erstellen und zu verwalten:

- CreateMultiRegionAccessPoint
- DescribeMultiRegionAccessPointOperation
- GetMultiRegionAccessPoint
- GetMultiRegionAccessPointPolicy
- GetMultiRegionAccessPointPolicyStatus
- GetMultiRegionAccessPointRoutes
- ListMultiRegionAccessPoints
- PutMultiRegionAccessPointPolicy
- SubmitMultiRegionAccessPointRoutes

Anzeigen Ihrer Weiterleitungskonfiguration für Multi-Region Access Points

AWS CLI

Mit dem folgenden Beispielbefehl wird Ihre Routenkonfiguration für Multi-Region Access Points abgerufen, sodass Sie die aktuellen Weiterleitungstatus für Ihre Buckets sehen können. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control get-multi-region-access-point-routes
--region eu-west-1
--account-id 111122223333
--mrap arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap
```

SDK for Java

Mit dem folgenden Code von SDK für Java wird Ihre Routenkonfiguration für Multi-Region Access Points abgerufen, sodass Sie die aktuellen Weiterleitungstatus für Ihre Buckets sehen können. Wenn Sie diese Beispielsyntax verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
S3ControlClient s3ControlClient = S3ControlClient.builder()
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider)
    .build();

GetMultiRegionAccessPointRoutesRequest request =
 GetMultiRegionAccessPointRoutesRequest.builder()
    .accountId("111122223333")
    .mrap("arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap")
    .build();

GetMultiRegionAccessPointRoutesResponse response =
 s3ControlClient.getMultiRegionAccessPointRoutes(request);
```

SDK for JavaScript

Das folgende SDK für JavaScript Code ruft Ihre Routenkonfiguration für Multi-Region Access Points ab, sodass Sie die aktuellen Weiterleitungsstatus für Ihre Buckets sehen können. Wenn Sie diese Beispielsyntax verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
const REGION = 'us-east-1'

const s3ControlClient = new S3ControlClient({
  region: REGION
})

export const run = async () => {
  try {
    const data = await s3ControlClient.send(
      new GetMultiRegionAccessPointRoutesCommand({
        AccountId: '111122223333',
        Mrap: 'arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap',
      })
    )
```

```
    console.log('Success', data)
    return data
  } catch (err) {
    console.log('Error', err)
  }
}

run()
```

SDK for Python

Mit dem folgenden Code von SDK für Python wird Ihre Routenkonfiguration für Multi-Region Access Points abgerufen, sodass Sie die aktuellen Weiterleitungstatus für Ihre Buckets sehen können. Wenn Sie diese Beispielsyntax verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
s3.get_multi_region_access_point_routes(
        AccountId=111122223333,
        Mrap=arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap)['Routes']
```

Aktualisieren der zugrunde liegende Bucket-Richtlinie von Amazon S3

Damit ein ordnungsgemäßer Zugriff gewährt wird, müssen Sie auch die zugrunde liegende Bucket-Richtlinie von Amazon S3 aktualisieren. In den folgenden Beispielen wird die Zugriffskontrolle an die Richtlinie für Multi-Region Access Points delegiert. Nachdem Sie die Zugriffskontrolle an die Richtlinie für Multi-Region Access Points delegiert haben, wird die Bucket-Richtlinie nicht mehr für die Zugriffskontrolle verwendet, wenn Anforderungen über den Multi-Region Access Point gestellt werden.

Hier ist ein Beispiel für eine Bucket-Richtlinie, die die Zugriffskontrolle an die Richtlinie der Multi-Region Access Points delegiert. Wenn Sie diese Bucket-Beispielrichtlinie verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen. Um diese Richtlinie über den AWS CLI `put-bucket-policy` Befehl anzuwenden, wie im nächsten Beispiel gezeigt, speichern Sie die Richtlinie in einer Datei, z. B. `policy.json`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": { "AWS": "*" },
```

```
    "Effect": "Allow",
    "Action": ["s3:*"],
    "Resource": ["arn:aws:s3:::111122223333/*", "arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
    "Condition": {
      "StringEquals": {
        "s3:DataAccessPointAccount": "444455556666"
      }
    }
  }
}
```

Der folgende Beispielbefehl `put-bucket-policy` verknüpft die aktualisierte S3-Bucket-Richtlinie mit Ihrem S3-Bucket:

```
aws s3api put-bucket-policy
  --bucket DOC-EXAMPLE-BUCKET
  --policy file:///tmp/policy.json
```

Aktualisieren einer Weiterleitungskonfiguration von Multi-Region Access Points

Mit dem folgenden Beispielbefehl wird die Weiterleitungskonfiguration des Multi-Region Access Points aktualisiert. Weiterleitungsbefehle für den Multi-Region Access Point können in einer dieser fünf Regionen ausgeführt werden:

- `ap-southeast-2`
- `ap-northeast-1`
- `us-east-1`
- `us-west-2`
- `eu-west-1`

In einer Weiterleitungskonfiguration für den Multi-Region Access Point können Sie Buckets auf einen aktiven oder passiven Weiterleitungsstatus festlegen. Aktive Buckets empfangen Traffic, passive Buckets nicht. Sie können den Weiterleitungsstatus eines Buckets festlegen, indem Sie den Wert `TrafficDialPercentage` für den Bucket auf `100` für aktiv oder `0` für passiv einstellen.

AWS CLI

Mit dem folgenden Beispielbefehl wird die Weiterleitungskonfiguration Ihres Multi-Region Access Points aktualisiert. In diesem Beispiel ist *DOC-EXAMPLE-BUCKET1* auf den aktiven Status und

DOC-EXAMPLE-BUCKET2 auf den passiven Status eingestellt. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control submit-multi-region-access-point-routes
--region ap-southeast-2
--account-id 111122223333
--mrap arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap
--route-updates Bucket=DOC-EXAMPLE-BUCKET1,TrafficDialPercentage=100
                Bucket=DOC-EXAMPLE-BUCKET2,TrafficDialPercentage=0
```

SDK for Java

Mit dem folgenden Code von SDK für Java wird die Weiterleitungskonfiguration Ihres Multi-Region Access Points aktualisiert. Wenn Sie diese Beispielsyntax verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
S3ControlClient s3ControlClient = S3ControlClient.builder()
    .region(Region.ap-southeast-2)
    .credentialsProvider(credentialsProvider)
    .build();

SubmitMultiRegionAccessPointRoutesRequest request =
 SubmitMultiRegionAccessPointRoutesRequest.builder()
    .accountId("111122223333")
    .mrap("arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap")
    .routeUpdates(
        MultiRegionAccessPointRoute.builder()
            .region("eu-west-1")
            .trafficDialPercentage(100)
            .build(),
        MultiRegionAccessPointRoute.builder()
            .region("ca-central-1")
            .bucket("111122223333")
            .trafficDialPercentage(0)
            .build()
    )
    .build();

SubmitMultiRegionAccessPointRoutesResponse response =
 s3ControlClient.submitMultiRegionAccessPointRoutes(request);
```

SDK for JavaScript

Mit dem folgenden SDK für JavaScript Code wird Ihre Weiterleitungskonfiguration für Multi-Region Access Points aktualisiert. Wenn Sie diese Beispielsyntax verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
const REGION = 'ap-southeast-2'

const s3ControlClient = new S3ControlClient({
  region: REGION
})

export const run = async () => {
  try {
    const data = await s3ControlClient.send(
      new SubmitMultiRegionAccessPointRoutesCommand({
        AccountId: '111122223333',
        Mrap: 'arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap',
        RouteUpdates: [
          {
            Region: 'eu-west-1',
            TrafficDialPercentage: 100,
          },
          {
            Region: 'ca-central-1',
            Bucket: 'DOC-EXAMPLE-BUCKET1',
            TrafficDialPercentage: 0,
          },
        ],
      })
    )
    console.log('Success', data)
    return data
  } catch (err) {
    console.log('Error', err)
  }
}

run()
```

SDK for Python

Mit dem folgenden Code von SDK für Python wird die Weiterleitungskonfiguration Ihres Multi-Region Access Points aktualisiert. Wenn Sie diese Beispielsyntax verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
s3.submit_multi_region_access_point_routes(
        AccountId=111122223333,
        Mrap=arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap,
        RouteUpdates= [{
            'Bucket': DOC-EXAMPLE-BUCKET,
            'Region': ap-southeast-2,
            'TrafficDialPercentage': 10
        }])
```

Hinzufügen eines Objekts zu einem Bucket in Ihrem Multi-Region Access Point

Verwenden Sie die Operaton PutObject, um ein Objekt zu dem Bucket hinzuzufügen, der dem Multi-Region Access Point zugeordnet ist. Aktivieren Sie die regionsübergreifende Replikation, um alle Buckets im Multi-Region Access Point synchron zu halten.

> **Note**
>
> Für diese Operation müssen Sie über die `s3:PutObject`-Berechtigung für den Multi-Region Access Point verfügen. Weitere Informationen zu den Berechtigungsanforderungen für Multi-Region Access Points finden Sie unter Berechtigungen.

AWS CLI

Das folgende Beispiel für eine Anforderung auf Datenebene lädt *example.txt* auf den angegebenen Multi-Region Access Point hoch. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3api put-object --bucket
 arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap --key example.txt --
body example.txt
```

SDK for Java

```
S3Client s3Client = S3Client.builder()
        .build();

PutObjectRequest objectRequest = PutObjectRequest.builder()
        .bucket("arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap")
        .key("example.txt")
        .build();

s3Client.putObject(objectRequest, RequestBody.fromString("Hello S3!"));
```

SDK for JavaScript

```
const client = new S3Client({});

async function putObjectExample() {
    const command = new PutObjectCommand({
        Bucket: "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap",
        Key: "example.txt",
        Body: "Hello S3!",
    });

    try {
        const response = await client.send(command);
        console.log(response);
    } catch (err) {
        console.error(err);
    }
}
```

SDK for Python

```
import boto3

client = boto3.client('s3')
client.put_object(
    Bucket='arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap',
    Key='example.txt',
    Body='Hello S3!'
)
```

Abrufen von Objekten von Ihrem Multi-Region Access Point

Verwenden Sie die Operation GetObject, um Objekte aus dem Multi-Region Access Point abzurufen.

> **Note**
>
> Für diese API-Operation müssen Sie über die `s3:GetObject`-Berechtigung für den Multi-Region Access Point verfügen. Weitere Informationen zu den Berechtigungsanforderungen für Multi-Region Access Points finden Sie unter Berechtigungen.

AWS CLI

Das folgende Beispiel für eine Anforderung auf Datenebene ruft die Datei *example.txt* vom angegebenen Multi-Region Access Point ab und lädt sie mit dem Namen *downloaded_example.txt* herunter. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3api get-object --bucket
 arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap --
key example.txt downloaded_example.txt
```

SDK for Java

```
S3Client s3 = S3Client
   .builder()
   .build();

GetObjectRequest getObjectRequest = GetObjectRequest.builder()
    .bucket("arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap")
    .key("example.txt")
    .build();

s3Client.getObject(getObjectRequest);
```

SDK for JavaScript

```
const client = new S3Client({})

async function getObjectExample() {
    const command = new GetObjectCommand({
```

```
        Bucket: "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap",
        Key: "example.txt"
    });

    try {
        const response = await client.send(command);
        console.log(response);
    } catch (err) {
        console.error(err);
    }
}
```

SDK for Python

```
import boto3

client = boto3.client('s3')
client.get_object(
    Bucket='arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap',
    Key='example.txt'
)
```

Auflisten von Objekten, die in einem Bucket gespeichert sind, der Ihrem Multi-Region Access Point zugrunde liegt

Verwenden Sie die Operation ListObjectsV2, um eine Liste von Objekten zurückzugeben, die in einem Bucket gespeichert sind, der Ihrem Multi-Region Access Point zugrunde liegt. Mit dem folgenden Beispielbefehl werden alle Objekte für den angegebenen Multi-Region Access Point aufgeführt, indem der ARN für den Multi-Region Access Point verwendet wird. In diesem Fall lautet der ARN für den Multi-Region Access Point:

`arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap`

> **Note**
>
> Für diese API-Operation müssen Sie über die `s3:ListBucket`-Berechtigung für den Multi-Region Access Point und den zugrunde liegenden Bucket verfügen. Weitere Informationen zu den Berechtigungsanforderungen für Multi-Region Access Points finden Sie unter Berechtigungen.

AWS CLI

In der folgenden Beispielanforderung auf Datenebene werden die Objekte in dem Bucket aufgeführt, der dem vom ARN angegebenen Multi-Region Access Point zugrunde liegt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3api list-objects-v2 --bucket
 arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap
```

SDK for Java

```
S3Client s3Client = S3Client.builder()
        .build();

String bucketName = "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap";

ListObjectsV2Request listObjectsRequest = ListObjectsV2Request
    .builder()
    .bucket(bucketName)
    .build();

 s3Client.listObjectsV2(listObjectsRequest);
```

SDK for JavaScript

```
const client = new S3Client({});

async function listObjectsExample() {
    const command = new ListObjectsV2Command({
        Bucket: "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap",
    });

    try {
        const response = await client.send(command);
        console.log(response);
    } catch (err) {
        console.error(err);
    }
}
```

SDK for Python

```
import boto3

client = boto3.client('s3')
client.list_objects_v2(
    Bucket='arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap'
)
```

Verwenden einer vorsignierte URL mit Multi-Region Access Points

Sie können eine vorsignierte URL verwenden, um eine URL zu generieren, die es anderen ermöglicht, über einen Amazon S3 Multi-Region Access Point auf Ihre Amazon-S3-Buckets zuzugreifen. Wenn Sie eine vorsignierte URL erstellen, verknüpfen Sie diese mit einer bestimmten Objektaktion, z. B. einem S3-Upload (`PutObject`) oder einem S3-Download (`GetObject`). Sie können die vorsignierte URL freigeben und jeder, der Zugriff darauf hat, kann die in die URL eingebettete Aktion so ausführen, als wäre er der ursprüngliche signierende Benutzer.

Vorsignierte URLs haben ein Ablaufdatum. Wenn das Ablaufdatum erreicht ist, funktioniert die URL nicht mehr.

Bevor Sie S3 Multi-Region Access Points mit vorsignierten URLs verwenden, überprüfen Sie die AWS -SDK-Kompatibilität mit dem SigV4A-Algorithmus. Stellen Sie sicher, dass Ihre SDK-Version SigV4A als Signaturimplementierung unterstützt, die zum Signieren der globalen Anforderungen von AWS-Region verwendet wird. Weitere Informationen über die Verwendung vorsignierter URLs in Amazon S3 finden Sie unter Gemeinsame Nutzung von Objekten unter Verwendung vorsignierter URLs.

Die folgenden Beispiele zeigen, wie Sie Multi-Region Access Points mit vorsignierten URLs verwenden können. Wenn Sie diese Beispiele verwenden möchten, ersetzen Sie die *`user input placeholders`* durch Ihre Informationen.

AWS CLI

```
aws s3 presign
 arn:aws:s3::123456789012:accesspoint/MultiRegionAccessPoint_alias/example-file.txt
```

SDK for Python

```
import logging
```

```
import boto3
from botocore.exceptions import ClientError

s3_client = boto3.client('s3',aws_access_key_id='xxx',aws_secret_access_key='xxx')
s3_client.generate_presigned_url(HttpMethod='PUT',ClientMethod="put_object",
 Params={'Bucket':'arn:aws:s3::123456789012:accesspoint/
abcdef0123456.mrap','Key':'example-file'})
```

SDK for Java

```
S3Presigner s3Presigner = S3Presigner.builder()
    .credentialsProvider(StsAssumeRoleCredentialsProvider.builder()
        .refreshRequest(assumeRole)
        .stsClient(stsClient)
        .build())
    .build();

GetObjectRequest getObjectRequest = GetObjectRequest.builder()
    .bucket("arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap")
    .key("example-file")
    .build();

GetObjectPresignRequest preSignedReq = GetObjectPresignRequest.builder()
    .getObjectRequest(getObjectRequest)
    .signatureDuration(Duration.ofMinutes(10))
    .build();

PresignedGetObjectRequest presignedGetObjectRequest =
 s3Presigner.presignGetObject(preSignedReq);
```

Note

Um SigV4A mit temporären Sicherheitsanmeldeinformationen zu verwenden, z. B. bei Verwendung von IAM-Rollen, stellen Sie sicher, dass Sie die temporären Anmeldeinformationen von einem regionalen Endpunkt in AWS Security Token Service (AWS STS) anstelle eines globalen Endpunkts anfordern. Wenn Sie den globalen Endpunkt für AWS STS (`sts.amazonaws.com`) verwenden, AWS STS generiert temporäre Anmeldeinformationen von einem globalen Endpunkt, der von Sig4A nicht unterstützt wird. Aus diesem Grund erhalten Sie eine Fehlermeldung. Um dieses Problem zu beheben, verwenden Sie einen der aufgelisteten regionalen Endpunkte für AWS STS.

Verwenden eines Buckets, der mit „Zahlung durch den Anforderer“ konfiguriert ist, mit Multi-Region Access Points

Wenn ein S3-Bucket, der Ihren Multi-Region Access Points zugeordnet ist, für die Verwendung von „Zahlung durch den Anforderer“ konfiguriert ist, trägt der Anforderer die Kosten für die Bucket-Anforderung, den Download und alle mit Multi-Region Access Points verbundenen Kosten. Weitere Informationen finden Sie unter Amazon S3 – Preise.

Hier sehen Sie ein Beispiel für eine Anforderung auf Datenebene an einen Multi-Region Access Point, der mit einem Bucket mit Zahlung durch den Anforderer verbunden ist.

AWS CLI

Wenn Sie Objekte von einem Multi-Region Access Point herunterladen möchten, der mit einem Bucket mit Zahlung durch den Anforderer verbunden ist, müssen Sie `--request-payer requester` als Teil Ihrer Anforderung get-object angeben. Sie müssen auch den Namen der Datei im Bucket sowie den Ort angeben, an dem sie gespeichert werden soll.

```
aws s3api get-object --bucket MultiRegionAccessPoint_ARN --request-payer requester
 --key example-file-in-bucket.txt example-location-of-downloaded-file.txt
```

SDK for Java

Wenn Sie Objekte von einem Multi-Region Access Point herunterladen möchten, der mit einem Bucket mit Zahlung durch den Anforderer verbunden ist, müssen Sie `RequestPayer.REQUESTER` als Teil Ihrer Anforderung `GetObject` angeben. Sie müssen auch den Namen der Datei in dem Bucket sowie den Ort angeben, an dem sie gespeichert werden soll.

```
GetObjectResponse getObjectResponse = s3Client.getObject(GetObjectRequest.builder()
    .key("example-file.txt")
    .bucket("arn:aws:s3::
123456789012:accesspoint/abcdef0123456.mrap")
    .requestPayer(RequestPayer.REQUESTER)
    .build()
).response();
```

Überwachung und Protokollierung von Anforderungen, die über einen Multi-Regions-Zugriffspunkt an zugrunde liegende Ressourcen erfolgen

Amazon S3 protokolliert Anforderungen, die über Multi-Region Access Points gestellt werden, und Anforderungen an die API-Operationen, die sie verwalten, wie z. B. `CreateMultiRegionAccessPoint` und `GetMultiRegionAccessPointPolicy`. Anforderungen, die über einen Multi-Regions-Zugriffspunkt an Amazon S3 gesendet werden, werden in den Zugriffsprotokollen und AWS CloudTrail-Protokollen des Amazon S3-Servers mit dem Hostnamen des Multi-Regions-Zugriffspunkts angezeigt. Der Hostname eines Zugriffspunkts hat das Format *`MRAP_alias`*`.accesspoint.s3-global.amazonaws.com`. Angenommen, Sie haben die folgende Bucket- und Multi-Regions-Zugriffspunktkonfiguration:

- Ein Bucket mit dem Namen `my-bucket-usw2` in der Region `us-west-2`, der das Objekt `my-image.jpg` enthält.
- Ein Bucket mit dem Namen `my-bucket-aps1` in der Region `ap-south-1`, der das Objekt `my-image.jpg` enthält.
- Ein Bucket mit dem Namen `my-bucket-euc1` in der Region `eu-central-1`, der kein Objekt namens `my-image.jpg` enthält.
- Ein Multi-Regions-Zugriffspunkt namens `my-mrap` mit dem Alias `mfzwi23gnjvgw.mrap`, der so konfiguriert ist, dass Anforderungen aus allen drei Buckets erfüllt werden.
- Ihre AWS-Konto-ID ist `123456789012`.

Eine Anforderung, die gesendet wird, um `my-image.jpg` direkt über einen der Buckets anzurufen, wird in Ihren Protokollen mit dem Hostnamen *`bucket_name`*`.s3.`*`Region`*`.amazonaws.com` angezeigt.

Wenn Sie die Anforderung stattdessen über den Multi-Region Access Point stellen, bestimmt Amazon S3 zunächst, welcher der Buckets in den verschiedenen Regionen am nächsten gelegen ist. Nach der Ermittlung, welcher Bucket zur Erfüllung der Anforderung verwendet werden soll, sendet Amazon S3 die Anforderung an diesen Bucket und protokolliert den Vorgang unter Verwendung des Hostnamens des Multi-Region Access Point. Wenn Amazon S3 in diesem Beispiel die Anforderung an `my-bucket-aps1` weiterleitet, spiegeln Ihre Protokolle eine erfolgreiche `GET`-Anforderung für `my-image.jpg` von `my-bucket-aps1` unter Verwendung des Hostnamens `mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com` wider.

> **Important**
>
> Multi-Region Access Points kennen den Dateninhalt der zugrunde liegenden Buckets nicht. Daher kann es sein, dass der Bucket, der die Anforderung erhält, die angeforderten Daten nicht enthält. Wenn Amazon S3 beispielsweise feststellt, dass der Bucket `my-bucket-euc1` am nächsten gelegen ist, spiegeln Ihre Protokolle eine fehlgeschlagene `GET`-Anforderung für `my-image.jpg` von `my-bucket-euc1` unter Verwendung des Hostnamens `mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com` wider. Wenn die Anforderung stattdessen an `my-bucket-usw2` weitergeleitet worden wäre, würden Ihre Protokolle eine erfolgreiche `GET`-Anforderung anzeigen.

Weitere Informationen zu Amazon S3-Server-Zugriffsprotokollen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung. Weitere Informationen zu AWS CloudTrail, finden Sie unter Was ist AWS CloudTrail? im AWS CloudTrail-Benutzerhandbuch.

Überwachen und Protokollieren von Anforderungen an Verwaltungs-API-Operationen von Multi-Region Access Points

Amazon S3 bietet verschiedene API-Operationen für die Verwaltung von Multi-Region Access Points, wie z. B. `CreateMultiRegionAccessPoint` und `GetMultiRegionAccessPointPolicy`. Wenn Sie Anforderungen an diese API-Operationen über die AWS Command Line Interface (AWS CLI), AWS-SDKs oder die Amazon-S3-REST-API stellen, verarbeitet Amazon S3 diese Anforderungen asynchron. Vorausgesetzt, Sie verfügen über die entsprechenden Berechtigungen für die Anforderung, gibt Amazon S3 ein Token für diese Anforderungen zurück. Sie können dieses Token mit `DescribeAsyncOperation` verwenden, um Ihnen zu helfen, den Status von laufenden asynchronen Vorgängen anzuzeigen. Amazon S3 verarbeitet `DescribeAsyncOperation`-Anforderungen synchron. Sie können die Amazon-S3-Konsole, die AWS CLI, SDKs oder die REST-API verwenden, um den Status asynchroner Anforderungen anzuzeigen.

> **Note**
>
> Die Konsole zeigt nur den Status asynchroner Anforderungen an, die innerhalb der letzten 14 Tage gestellt wurden. Verwende Sie AWS CLI, SDKs oder REST-API, um den Status älterer Anforderungen anzuzeigen.

Asynchrone Verwaltungsvorgänge können einen von mehreren Zuständen aufweisen:

NEW

Amazon S3 hat die Anforderung erhalten und bereitet die Ausführung des Vorgangs vor.

IN_PROGRESS

Amazon S3 führt derzeit den Vorgang aus.

SUCCESS

Der Vorgang war erfolgreich. Die Antwort enthält relevante Informationen, z. B. den Multi-Regions-Zugriffspunkt-Alias für eine `CreateMultiRegionAccessPoint`-Anforderung.

FAILED

Der Vorgang schlägt fehl. Die Antwort enthält eine Fehlermeldung, die den Grund für den Anforderungsfehler angibt.

Verwenden von AWS CloudTrail mit Multi-Region Access Points

Mit AWS CloudTrail können Sie Kontoaktivitäten in Ihrer AWS-Infrastruktur anzeigen, suchen, herunterladen, archivieren, analysieren und auf diese reagieren. Mit Multi-Region Access Points und CloudTrail-Protokollierung können Sie Folgendes ermitteln:

- Wer oder was welche Maßnahme ergriffen hat
- Welche Ressourcen in Anspruch genommen wurden
- Wann das Ereignis aufgetreten ist
- Weitere Details zu dem Ereignis

Sie können diese Protokollinformationen für die Analyse und Reaktion auf Aktivitäten verwenden, die über Ihre Multi-Region Access Points erfolgt sind.

Einrichten von AWS CloudTrail-Multi-Regions-Zugriffspunkten

Um die CloudTrail-Protokollierung für Vorgänge im Zusammenhang mit dem Erstellen oder Verwalten von Multi-Region Access Points zu aktivieren, müssen Sie die CloudTrail-Protokollierung so konfigurieren, dass die Ereignisse in der Region USA West (Oregon) aufgezeichnet werden. Sie müssen Ihre Protokollierungskonfiguration auf diese Weise einrichten, unabhängig davon, in welcher Region Sie sich befinden, wenn Sie die Anforderung stellen, oder welche Regionen der Multi-Region Access Point unterstützt. Alle Anforderungen, einen Multi-Region Access Point zu erstellen oder zu

verwalten, werden über die Region USA West (Oregon) geleitet. Wir empfehlen Ihnen, diese Region entweder einem vorhandenen Trail hinzuzufügen oder einen neuen Trail zu erstellen, der diese Region und alle Regionen enthält, die mit dem Multi-Region Access Point verknüpft sind.

Amazon S3 protokolliert alle Anforderungen, die über einen Multi-Region Access Point erfolgen, und Anforderungen an die API-Operationen, die Zugriffspunkte verwalten, z. B. `CreateMultiRegionAccessPoint` und `GetMultiRegionAccessPointPolicy`. Wenn Sie diese Anforderungen über einen Multi-Regions-Zugriffspunkt protokollieren, werden sie in Ihren AWS CloudTrail-Protokollen mit dem Hostnamen des Multi-Regions-Zugriffspunkts erscheinen. Wenn Sie beispielsweise Anforderungen an einen Bucket über einen Multi-Region Access Point mit dem Alias `mfzwi23gnjvgw.mrap` stellen, haben die Einträge im CloudTrail-Protokoll den Hostnamen `mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com`.

Multi-Region Access Points leiten Anforderungen an den nächstgelegenen Bucket weiter. Aufgrund dieses Verhaltens sind beim Betrachten der CloudTrail-Protokolle für einen Multi-Region Access Point Anforderungen an die zugrunde liegenden Buckets zu sehen. Einige dieser Anforderungen sind möglicherweise direkte Anforderungen an den Bucket, die nicht über den Multi-Region Access Point geleitet werden. Beachten Sie dies bei der Überprüfung des Datenverkehrs. Wenn sich ein Bucket in einem Multi-Regions-Zugriffspunkt befindet, können Anforderungen direkt an diesen Bucket gestellt werden, ohne den Multi-Regions-Zugriffspunkt zu durchlaufen.

Beim Erstellen und Verwalten von Multi-Regions-Zugriffspunkten sind asynchrone Ereignisse erforderlich. Asynchrone Anforderungen haben keine Abschlussereignisse im CloudTrail-Protokoll. Weitere Informationen zu asynchronen Anforderungen finden Sie unter Überwachen und Protokollieren von Anforderungen an Verwaltungs-API-Operationen von Multi-Region Access Points.

Weitere Informationen zu AWS CloudTrail, finden Sie unter Was ist AWS CloudTrail? im AWS CloudTrail-Benutzerhandbuch.

Amazon-S3-Sicherheit

Cloud-Sicherheit bei AWS hat höchste Priorität. Als - AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die entwickelt wurde, um die Anforderungen der sicherheitssensibelsten Organisationen zu erfüllen.

Sicherheit ist eine geteilte Verantwortung zwischen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

Sicherheit der Cloud

AWS ist für den Schutz der Infrastruktur verantwortlich, die - AWS Services in der ausführt AWS Cloud. stellt Ihnen AWS außerdem Services bereit, die Sie sicher nutzen können. Die Wirksamkeit unserer Sicherheitsfunktionen wird regelmäßig von externen Prüfern im Rahmen des AWS - Compliance-Programms getestet und überprüft. Weitere Informationen zu den für Amazon S3 geltenden Compliance-Programmen finden Sie unter AWS Im Rahmen des Compliance-Programms zugelassene -Services.

Sicherheit in der Cloud

Ihre Verantwortung wird durch den AWS Service bestimmt, den Sie verwenden. In Ihre Verantwortung fallen außerdem weitere Faktoren, wie z. B. die Vertraulichkeit der Daten, die Anforderungen Ihrer Organisation sowie geltende Gesetze und Vorschriften. Für Amazon S3 umfasst Ihre Verantwortlichkeit die folgenden Bereiche:

- Verwalten Ihrer Daten, einschließlich Objekt-Eigentumsrechte und Verschlüsselung.
- Klassifizierung Ihres Vermögens.
- Zugriffsverwaltung zu Ihren Daten unter Verwendung von IAM-Rollen und andere Dienstkonfigurationen, um die entsprechenden Berechtigungen anzuwenden.
- Aktivieren detektivischer Kontrollen wie AWS CloudTrail oder Amazon GuardDuty für Amazon S3.

Diese Dokumentation beschreibt, wie Sie das Modell der übergreifenden Verantwortlichkeit bei der Verwendung von Amazon S3 anwenden können. Die folgenden Themen veranschaulichen, wie Sie Amazon S3 zur Erfüllung Ihrer Sicherheits- und Compliance-Ziele konfigurieren können. Sie erfahren auch, wie Sie andere - AWS Services verwenden, die Sie bei der Überwachung und Sicherung Ihrer Amazon S3-Ressourcen unterstützen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Datenschutz in Amazon S3

Amazon S3 bietet eine sehr robuste Speicherinfrastruktur, die für geschäftskritische und primäre Speicheranwendungen entwickelt wurde. S3-Standard, S3 Intelligent-Tiering, S3-Standard-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive speichern Objekte redundant auf mehreren Geräten in mindestens drei Availability Zones in einem AWS-Region. Eine Availability Zone ist eines oder mehrere diskrete Rechenzentren mit redundanter Stromversorgung, Vernetzung und Konnektivität in einem AWS-Region. Availability Zones sind physisch durch eine bedeutende Entfernung von vielen Kilometern von jeder anderen Availability

Zone getrennt, obwohl alle innerhalb von 100 km (60 Meilen) voneinander liegen. Die Speicherklasse S3 One Zone-IA speichert Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone. Diese Services wurden entwickelt, um mit gleichzeitigen Geräteausfällen umzugehen, indem sie verlorene Redundanz schnell erkennen und reparieren, und auch regelmäßig die Integrität Ihrer Daten mithilfe von Prüfsummen überprüfen.

Der Amazon-S3-Standardspeicher bietet folgende Funktionen:

- Unterstützung durch Service Level Agreement von Amazon S3.
- Auf 99,999999999 %-ige Zuverlässigkeit und 99,99 %-ige Verfügbarkeit von Objekten über einen Zeitraum von einem Jahr ausgelegt.
- S3-Standard, S3 Intelligent-Tiering, S3-Standard-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive sind alle darauf ausgelegt, Daten im Falle des Verlustes einer gesamten Amazon-S3-Availability-Zone zu erhalten.

Darüber hinaus schützt Amazon S3 Ihre Daten durch Versioning. Sie können Versioning verwenden, um sämtliche Versionen aller Objekte in Ihrem Amazon-S3-Bucket zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leicht wiederherstellen. Standardmäßig wird durch Anforderungen die zuletzt geschriebene Version abgerufen. Ältere Versionen eines Objekts können abgerufen werden, indem die entsprechende Version des Objekts in der Anforderung angegeben wird.

Neben S3 Versioning können Sie auch Amazon S3 Object Lock und S3 Replication verwenden, um Ihre Daten zu schützen. Weitere Informationen finden Sie im Tutorial: Schutz von Daten in Amazon S3 vor versehentlichem Löschen oder Anwendungsfehlern mithilfe von S3 Versionierung, S3 Object Lock und S3 Replication.

Aus Datenschutzgründen empfehlen wir, die AWS-Konto Anmeldeinformationen zu schützen und individuelle Benutzerkonten mit einzurichten AWS Identity and Access Management, damit jeder Benutzer nur die Berechtigungen erhält, die er für seine Aufgaben benötigt.

Wenn Sie für den Zugriff auf AWS über eine Befehlszeilenschnittstelle oder eine API FIPS-140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2.

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls den Adressdatenschutz in Amazon S3:

- Implement server-side encryption
- Enforce encryption of data in transit
- Consider using Macie with Amazon S3
- Identify and audit all your Amazon S3 buckets
- Monitor Amazon Web Services security advisories

Datenschutz durch Verschlüsselung

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Dieser Datenschutz bezieht sich auf Daten bei der Übertragung (wenn sie zu Amazon S3 oder von diesem geschickt werden), ebenso wie auf ruhende Daten (die in Amazon-S3-Rechenzentren auf Datenträgern gespeichert sind). Sie können Daten während der Übertragung mit Secure Socket Layer/Transport Layer Security (SSL/TLS) oder clientseitiger Verschlüsselung schützen. Sie haben folgende Optionen, um Daten im Ruhezustand in Amazon S3 zu schützen:

- Serverseitige Verschlüsselung – Amazon S3 verschlüsselt Ihre Objekte, bevor sie auf Datenträgern in AWS Rechenzentren gespeichert werden, und entschlüsselt sie dann, wenn Sie sie herunterladen.

  Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert und alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden im Ruhezustand automatisch verschlüsselt. Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) ist die Standardverschlüsselungskonfiguration für jeden Bucket in Amazon S3. Um einen anderen Verschlüsselungstyp zu verwenden, können Sie entweder die Art der serverseitigen

Verschlüsselung angeben, die in Ihren S3-PUT-Anfragen verwendet werden soll, oder Sie können die Standardverschlüsselungskonfiguration im Ziel-Bucket festlegen.

Wenn Sie in Ihren PUT Anforderungen einen anderen Verschlüsselungstyp angeben möchten, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) oder die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden. Wenn Sie im Ziel-Bucket eine andere Standardverschlüsselungskonfiguration festlegen möchten, können Sie SSE-KMS oder DSSE-KMS verwenden.

Weitere Informationen zu den einzelnen Optionen für die serverseitige Verschlüsselung finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung.

Informationen zum Konfigurieren der serverseitigen Verschlüsselung finden Sie unter:

- Angeben serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)
- Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS)
- the section called “Angeben von DSSE-KMS”
- Angeben der serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

- Clientseitige Verschlüsselung – Sie können Daten clientseitig verschlüsseln und die verschlüsselten Daten auf Amazon S3 hochladen. In diesem Fall verwalten Sie den Verschlüsselungsprozess, die Verschlüsselungsschlüssel und die zugehörigen Tools.

  Informationen zum Konfigurieren der clientseitigen Verschlüsselung finden Sie unter Schützen von Daten mithilfe der clientseitigen Verschlüsselung.

Wenn Sie feststellen möchten, welcher Prozentsatz Ihrer Speicherbytes verschlüsselt ist, können Sie die Metriken von Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Weitere Informationen finden Sie unter Bewertung Ihrer Speicheraktivität und -nutzung mit S3 Storage Lens. Eine vollständige Liste der Metriken finden Sie im Glossar der S3-Storage-Lens-Metriken.

Weitere Informationen zur serverseitigen Verschlüsselung und zur clientseitigen Verschlüsselung finden Sie in den unten aufgeführten Themen.

Themen

Schützen von Daten mit serverseitiger Verschlüsselung

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Serverseitige Verschlüsselung ist die Verschlüsselung von Daten am Zielort durch die Anwendung oder den Service, der sie erhält. Amazon S3 verschlüsselt Ihre Daten auf Objektebene, während es sie auf Festplatten in AWS Rechenzentren schreibt, und entschlüsselt sie für Sie, wenn Sie darauf zugreifen. Wenn Sie Ihre Anforderung authentifizieren und Zugriffsberechtigungen besitzen, gibt es in Bezug auf die Art und Weise, wie Sie auf verschlüsselte oder nicht verschlüsselte Objekte zugreifen, keinen Unterschied. Wenn Sie beispielsweise Ihre Objekte unter Verwendung einer vorsignierten URL teilen, verhält sich die URL für verschlüsselte und unverschlüsselte Objekte gleich. Wenn Sie die Objekte in Ihrem Bucket auflisten, gibt die Listen-API-Operation außerdem eine Liste aller Objekte zurück, unabhängig davon, ob sie verschlüsselt sind.

Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert und alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden im Ruhezustand automatisch verschlüsselt. Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) ist die Standardverschlüsselungskonfiguration für jeden Bucket in Amazon S3. Um einen anderen Verschlüsselungstyp zu verwenden, können Sie entweder die Art der serverseitigen Verschlüsselung angeben, die in Ihren S3-PUT-Anfragen verwendet werden soll, oder Sie können die Standardverschlüsselungskonfiguration im Ziel-Bucket festlegen.

Wenn Sie in Ihren PUT Anforderungen einen anderen Verschlüsselungstyp angeben möchten, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) oder die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden. Wenn Sie im Ziel-Bucket eine andere Standardverschlüsselungskonfiguration festlegen möchten, können Sie SSE-KMS oder DSSE-KMS verwenden.

> **Note**
>
> Sie können nicht gleichzeitig unterschiedliche Arten serverseitiger Verschlüsselung auf dasselbe Objekt anwenden.

Wenn Sie Ihre vorhandenen Objekte verschlüsseln müssen, verwenden Sie S3 Batch Operations und S3 Inventory. Weitere Informationen finden Sie unter Verschlüsseln von Objekten mit Amazon S3 Batch Operations) und Ausführung umfangreicher Batch-Vorgänge für Amazon S3-Objekte durch..

Sie haben vier sich gegenseitig ausschließende Optionen für die serverseitige Verschlüsselung, abhängig davon, wie Sie die Verschlüsselungsschlüssel verwalten und wie viele Verschlüsselungsebenen Sie anwenden möchten.

Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)

Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert. Die Standardoption für die serverseitige Verschlüsselung besteht in von Amazon S3 verwalteten Schlüsseln (SSE-S3). Jedes Objekt wird mit einem eindeutigen Schlüssel verschlüsselt. Als zusätzliche Sicherheit verschlüsselt SSE-S3 den Schlüssel selbst mit einem Root-Schlüssel, der regelmäßig rotiert. SSE-S3 verwendet für die Verschlüsselung Ihrer Daten eine der stärksten verfügbaren Blockverschlüsselungen: 256-bit Advanced Encryption Standard (AES-256). Weitere Informationen finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

Serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS)

Die serverseitige Verschlüsselung mit AWS KMS keys (SSE-KMS) wird durch eine Integration des AWS KMS Service mit Amazon S3 bereitgestellt. Mit haben AWS KMS Sie mehr Kontrolle über Ihre Schlüssel. Sie können beispielsweise separate Schlüssel anzeigen, Kontrollrichtlinien bearbeiten und den Schlüsseln in AWS CloudTrail folgen. Darüber hinaus können Sie vom Kunden verwaltete

Schlüssel erstellen und verwalten oder von Von AWS verwaltete Schlüssel verwaltete Schlüssel verwenden, die für Sie, Ihren Service und Ihre Region einzigartig sind. Weitere Informationen finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

Serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS)

Die serverseitige Dual-Layer-Verschlüsselung mit AWS KMS keys (DSSE-KMS) ähnelt SSE-KMS, aber DSSE-KMS wendet zwei einzelne Verschlüsselungsebenen auf Objektebene anstelle einer Ebene an. Da beide Verschlüsselungsebenen auf ein Objekt auf der Serverseite angewendet werden, können Sie eine Vielzahl von - AWS-Services und -Tools verwenden, um Daten in S3 zu analysieren, während Sie eine Verschlüsselungsmethode verwenden, die Ihre Compliance-Anforderungen erfüllen kann. Weitere Informationen finden Sie unter Verwenden der serverseitigen Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS).

Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Bei serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwalten Sie die Verschlüsselungsschlüssel und Amazon S3 verwaltet die Verschlüsselung, wenn es auf Festplatten schreibt, und die Entschlüsselung, wenn Sie auf Ihre Objekte zugreifen. Weitere Informationen finden Sie unter Verwenden von serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C).

Amazon S3 verschlüsselt jetzt automatisch alle neuen Objekte

Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. SSE-S3, das den 256-Bit Advanced Encryption Standard (AES-256) verwendet, wird automatisch auf alle neuen Buckets und auf alle vorhandenen S3-Buckets angewendet, für die noch keine Standardverschlüsselung konfiguriert ist. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface (AWS CLI) und den - AWS SDKs verfügbar.

In den folgenden Abschnitten werden Fragen zu diesem Update beantwortet.

Ändert Amazon S3 die Standardverschlüsselungseinstellungen für meine vorhandenen Buckets, für die bereits die Standardverschlüsselung konfiguriert ist?

Nein. Es gibt keine Änderungen an der Standardverschlüsselungskonfiguration für einen vorhandenen Bucket, für den bereits SSE-S3 oder serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) konfiguriert ist. Weitere Informationen zum Festlegen des Standardverschlüsselungsverhalten für Buckets finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets. Weitere Informationen zu den Verschlüsselungseinstellungen von SSE-S3 und SSE-KMS finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung.

Wird die Standardverschlüsselung für meine vorhandenen Buckets aktiviert, für die keine Standardverschlüsselung konfiguriert ist?

Ja. Amazon S3 konfiguriert jetzt die Standardverschlüsselung für alle vorhandenen unverschlüsselten Buckets für die serverseitige Verschlüsselung mit S3-verwalteten Schlüsseln (SSE-S3) als Basisverschlüsselungsstufe für neue Objekte, die in diese Buckets hochgeladen werden. Objekte, die sich bereits in einem vorhandenen nicht verschlüsselten Bucket befinden, werden nicht automatisch verschlüsselt.

Wie kann ich den Standardverschlüsselungsstatus neuer Objekt-Uploads einsehen?

Derzeit können Sie den Standardverschlüsselungsstatus neuer Objekt-Uploads in - AWS CloudTrail Protokollen, S3 Inventory und S3 Storage Lens, der Amazon S3-Konsole und als zusätzlichen Amazon S3-API-Antwort-Header in der AWS Command Line Interface (AWS CLI) und den - AWS SDKs anzeigen.

- Informationen zum Anzeigen Ihrer CloudTrail Ereignisse finden Sie unter Anzeigen von CloudTrail Ereignissen in der CloudTrail Konsole im AWS CloudTrail -Benutzerhandbuch. - CloudTrail Protokolle bieten API-Nachverfolgung für - `PUT` und -`POST`Anfragen an Amazon S3. Wenn die Standardverschlüsselung zum Verschlüsseln von Objekten in Ihren Buckets verwendet wird, enthalten die CloudTrail Protokolle für - `PUT` und -`POST`API-Anforderungen das folgende Feld als Name-Wert-Paar: `"SSEApplied":"Default_SSE_S3"`.
- Um den automatischen Verschlüsselungsstatus neuer Objektuploads in S3 Inventar anzuzeigen, konfigurieren Sie einen S3-Inventarbericht, der das Feld Verschlüsselungsmetadaten enthält, und sehen Sie sich dann den Verschlüsselungsstatus jedes neuen Objekts im Bericht an. Weitere Informationen finden Sie unter Einrichtung von Amazon S3 Inventory.
- Um den automatischen Verschlüsselungsstatus für neue Objektuploads in S3 Storage Lens anzuzeigen, konfigurieren Sie ein S3-Storage-Lens-Dashboard und sehen Sie sich die Metriken

Verschlüsselte Bytes und Anzahl verschlüsselter Objekte in der Kategorie Datenschutz des Dashboards an. Weitere Informationen finden Sie unter Erstellen eines Amazon S3-Storage-Lens-Dashboards und Anzeigen von S3-Storage-Lens-Metriken in den Dashboards.

- Wenn Sie den Status der automatischen Verschlüsselung auf Bucket-Ebene in der Amazon-S3-Konsole anzeigen möchten, überprüfen Sie die Standardverschlüsselung Ihrer Amazon-S3-Buckets in der Amazon-S3-Konsole. Weitere Informationen finden Sie unter Konfigurieren der Standardverschlüsselung.
- Um den automatischen Verschlüsselungsstatus als zusätzlichen Amazon S3-API-Antwort-Header in der AWS Command Line Interface (AWS CLI) und den - AWS SDKs anzuzeigen, überprüfen Sie den Antwort-Header, `x-amz-server-side-encryption` wenn Sie Objektaktions-APIs wie PutObject und verwendenGetObject.

Was muss ich tun, um von dieser Änderung zu profitieren?

Sie müssen keine Änderungen an Ihren bestehenden Anwendungen vornehmen. Da die Standardverschlüsselung für alle Ihre Buckets aktiviert ist, werden alle neuen Objekte, die auf Amazon S3 hochgeladen werden, automatisch verschlüsselt.

Kann ich die Verschlüsselung für die neuen Objekte deaktivieren, die in meinen Bucket geschrieben werden?

Nein. SSE-S3 ist die neue Basisverschlüsselungsstufe, die auf alle neuen Objekte angewendet wird, die in Ihren Bucket hochgeladen werden. Sie können die Verschlüsselung für das Hochladen neuer Objekte nicht mehr deaktivieren.

Werden sich meine Gebühren ändern?

Nein. Die Standardverschlüsselung mit SSE-S3 ist ohne zusätzliche Kosten verfügbar. Speicherplatz, Anfragen und andere S3-Funktionen werden Ihnen wie gewohnt in Rechnung gestellt. Preisinformationen finden Sie unter Amazon S3 – Preise.

Verschlüsselt Amazon S3 meine vorhandenen unverschlüsselten Objekte?

Nein. Ab dem 5. Januar 2023 verschlüsselt Amazon S3 nur neue Objekt-Uploads automatisch. Zum Verschlüsseln bestehender Objekte können Sie S3-Batch-Operationen verwenden, um verschlüsselte Kopien Ihrer Objekte zu erstellen. Diese verschlüsselten Kopien behalten die vorhandenen Objektdaten und den Namen bei und werden mit den von Ihnen angegebenen Verschlüsselungsschlüsseln verschlüsselt. Weitere Informationen finden Sie unter Encrypting objects with Amazon S3 Batch Operations im AWS Storage Blog.

Ich habe vor dieser Version die Verschlüsselung für meine Buckets nicht aktiviert. Muss ich die Art und Weise, wie ich auf Objekte zugreife, ändern?

Nein. Die Standardverschlüsselung mit SSE-S3 verschlüsselt Ihre Daten automatisch, während sie in Amazon S3 geschrieben werden, und entschlüsselt sie für Sie, wenn Sie darauf zugreifen. Die Art und Weise, wie Sie auf automatisch verschlüsselte Objekte zugreifen, ändert sich nicht.

Muss ich die Art und Weise, wie ich auf meine clientseitig verschlüsselten Objekte zugreife, ändern?

Nein. Alle clientseitig verschlüsselten Objekte, die vor dem Hochladen in Amazon S3 verschlüsselt werden, kommen als verschlüsselte Geheimtextobjekte in Amazon S3 an. Diese Objekte verfügen jetzt über eine zusätzliche SSE-S3-Verschlüsselungsebene. Ihre Workloads, die clientseitig verschlüsselte Objekte verwenden, erfordern keine Änderungen an Ihren Client-Services oder Autorisierungseinstellungen.

> **Note**
>
> HashiCorp Terraform-Benutzer, die keine aktualisierte Version des AWS Anbieters verwenden, sehen möglicherweise eine unerwartete Abweichung, nachdem sie neue S3-Buckets ohne vom Kunden definierte Verschlüsselungskonfiguration erstellt haben. Um diese Abweichung zu vermeiden, aktualisieren Sie Ihre Terraform- AWS Provider-Version auf eine der folgenden Versionen: eine beliebige 4.x Version, 3.76.1oder 2.70.4.

Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Alle neuen Objekt-Uploads in Amazon-S3-Buckets werden standardmäßig mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt.

Die serverseitige Verschlüsselung schützt Daten im Ruhezustand. Amazon S3 verschlüsselt jedes Objekt mit einem eindeutigen Schlüssel. Als zusätzliche Sicherheit verschlüsselt es den Schlüssel selbst mit einem Schlüssel, der regelmäßig rotiert. Die serverseitige Amazon-S3-Verschlüsselung verwendet den 256-Bit-Advanced-Encryption-Standard-Galois/Counter-Mode (AES-GCM), um alle hochgeladenen Objekte zu verschlüsseln.

Es fallen keine weiteren Gebühren für die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) an. Für Anfragen zum Konfigurieren der Standard-Verschlüsselungsfunktion werden jedoch Standardgebühren für Amazon-S3-Anfragen berechnet. Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

Wenn Sie möchten, dass Ihre Daten-Uploads ausschließlich mit von Amazon S3 verwalteten Schlüsseln verschlüsselt werden, können Sie die folgende Bucket-Richtlinie verwenden. Beispielsweise verweigert die folgende Bucket-Richtlinie Berechtigungen zum Hochladen von Objekten, wenn die Anforderung nicht den `x-amz-server-side-encryption`-Header enthält, der eine serverseitige Verschlüsselung anfordert:

```
{
  "Version": "2012-10-17",
  "Id": "PutObjectPolicy",
  "Statement": [
    {
      "Sid": "DenyObjectsThatAreNotSSES3",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
      "Condition": {
        "StringNotEquals": {
          "s3:x-amz-server-side-encryption": "AES256"
        }
      }
    }
   ]
}
```

> **Note**
>
> Die serverseitige Verschlüsselung verschlüsselt nur die Objektdaten, nicht die Metadaten des Objekts.

API-Support für die serverseitige Verschlüsselung

Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert und alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden im Ruhezustand automatisch verschlüsselt. Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) ist die Standardverschlüsselungskonfiguration für jeden Bucket in Amazon S3. Um einen anderen Verschlüsselungstyp zu verwenden, können Sie entweder die Art der serverseitigen Verschlüsselung angeben, die in Ihren S3-PUT-Anfragen verwendet werden soll, oder Sie können die Standardverschlüsselungskonfiguration im Ziel-Bucket festlegen.

Wenn Sie in Ihren PUT Anforderungen einen anderen Verschlüsselungstyp angeben möchten, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) oder die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden. Wenn Sie im Ziel-Bucket eine andere Standardverschlüsselungskonfiguration festlegen möchten, können Sie SSE-KMS oder DSSE-KMS verwenden.

Zum Konfigurieren der serverseitigen Verschlüsselung mit den REST-APIs für die Objekterstellung müssen Sie den Anforderungs-Header `x-amz-server-side-encryption` bereitstellen. Weitere Information zu den REST-APIs finden Sie unter Verwenden der REST-API.

Die folgenden Amazon-S3-APIs unterstützen diesen Header:

- PUT-Operationen – Geben Sie den Anforderungs-Header an, wenn Sie Daten mithilfe der PUT-API hochladen. Weitere Informationen finden Sie unter PUT Object.
- Mehrteiligen Upload initiieren – Geben Sie den Header in der Initiierungsanforderung an, wenn Sie große Objekte mit der API für mehrteilige Uploads hochladen. Weitere Informationen finden Sie unter Mehrteiligen Upload initiieren.
- COPY-Operationen – Wenn Sie ein Objekt kopieren, erhalten Sie ein Quell- und ein Zielobjekt. Weitere Informationen finden Sie unter PUT Object – Copy.

> **Note**
>
> Wenn Sie eine POST-Operation für das Hochladen eines Objekts verwenden, anstatt den Anforderungs-Header anzugeben, stellen Sie die gleichen Informationen in den Formularfeldern bereit. Weitere Informationen finden Sie unter POST Object.

Die AWS SDKs stellen auch Wrapper-APIs bereit, mit denen Sie eine serverseitige Verschlüsselung anfordern können. Sie können auch die verwenden AWS Management Console , um Objekte hochzuladen und eine serverseitige Verschlüsselung anzufordern.

Weitere Informationen finden Sie unter AWS KMS -Konzepte im Entwicklerhandbuch zu AWS Key Management Service .

Themen

- Angeben serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)

Angeben serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert und alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden im Ruhezustand automatisch verschlüsselt. Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) ist die Standardverschlüsselungskonfiguration für jeden Bucket in Amazon S3. Um einen anderen Verschlüsselungstyp zu verwenden, können Sie entweder die Art der serverseitigen

Verschlüsselung angeben, die in Ihren S3-PUT-Anfragen verwendet werden soll, oder Sie können die Standardverschlüsselungskonfiguration im Ziel-Bucket festlegen.

Wenn Sie in Ihren PUT Anforderungen einen anderen Verschlüsselungstyp angeben möchten, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) oder die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden. Wenn Sie im Ziel-Bucket eine andere Standardverschlüsselungskonfiguration festlegen möchten, können Sie SSE-KMS oder DSSE-KMS verwenden.

Sie können SSE-S3 mithilfe der S3-Konsole, der REST-APIs , der AWS SDKs und AWS Command Line Interface () angeben AWS CLI. Weitere Informationen finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

Verwenden der S3-Konsole

In diesem Thema wird beschrieben, wie der Verschlüsselungstyp eines Objekts mit der AWS Management Console festgelegt oder geändert wird. Wenn Sie ein Objekt unter Verwendung der Konsole kopieren, kopiert Amazon S3 das Objekt unverändert. Wenn das Quellobjekt verschlüsselt ist, wird das Zielobjekt demnach ebenfalls verschlüsselt. Sie können mit der Konsole die Verschlüsselung für ein Objekt hinzufügen oder ändern.

> **Note**
>
> Wenn Sie die Verschlüsselung eines Objekts ändern, wird ein neues Objekt erstellt, um das alte zu ersetzen. Wenn S3-Versioning aktiviert ist, wird eine neue Version des Objekts erstellt, und das vorhandene Objekt wird zu einer älteren Version. Die Rolle, die die Eigenschaft ändert, wird auch Besitzer des neuen Objekts (oder der neuen Objektversion).

So ändern Sie die Verschlüsselung für ein Objekt

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
4. Wählen Sie in der Liste Objekte den Namen des Objekts aus, für das Sie eine Verschlüsselung hinzufügen oder ändern möchten.

Die Detailseite des Objekts wird angezeigt. Sie enthält mehrere Abschnitte mit den Eigenschaften des Objekts.

5. Wählen Sie die Registerkarte Eigenschaften aus.
6. Scrollen Sie nach unten zum Abschnitt Serverseitige Verschlüsselungseinstellungen und wählen Sie dann Bearbeiten aus.
7. Wählen Sie unter Verschlüsselungseinstellungen die Option Verwenden von Bucket-Einstellungen für die Standardverschlüsselung oder Überschreiben der Bucket-Einstellungen für die Standardverschlüsselung aus.
8. Wenn Sie Überschreiben der Bucket-Einstellungen für die Standardverschlüsselung ausgewählt haben, konfigurieren Sie die folgenden Verschlüsselungseinstellungen.

   - Wählen Sie unter Verschlüsselungstyp die Option Von Amazon S3 verwaltete Schlüssel (SSE-S3) aus. SSE-S3 verwendet für die Verschlüsselung der einzelnen Objekte eine der stärksten Blockverschlüsselungen: 256-bit Advanced Encryption Standard (AES-256). Weitere Informationen finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

9. Wählen Sie Save Changes (Änderungen speichern).

> **Note**
>
> Diese Aktion wendet auf alle angegebenen Objekte Verschlüsselung an. Warten Sie beim Verschlüsseln von Ordnern, bis die Speicheroperation abgeschlossen ist, bevor Sie dem Ordner neue Objekte hinzufügen.

Verwenden der REST-API

Wenn das Objekt erstellt wird – d. h. wenn Sie ein neues Objekt hochladen oder eine Kopie eines vorhandenen Objekts anlegen –, können Sie angeben, ob Amazon S3 Ihre Daten mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsseln soll, indem Sie der Anforderung den Header `x-amz-server-side-encryption` hinzufügen. Setzen Sie den Wert des Headers auf den Verschlüsselungsalgorithmus `AES256`, den Amazon S3 unterstützt. Amazon S3 bestätigt, dass Ihr Objekt unter Verwendung von SSE-S3 gespeichert wird, indem der Antwort-Header `x-amz-server-side-encryption` zurückgegeben wird.

Die folgenden Upload-REST-API-Operationen akzeptieren den Anforderungs-Header `x-amz-server-side-encryption`.

- PUT Object
- PUT Object – Kopieren
- POST Object
- Initiieren eines mehrteiligen Uploads

Wenn Sie große Objekte mit der API für mehrteilige Uploads hochladen, können Sie die serverseitige Verschlüsslung festlegen, indem Sie der Anforderung zum Initiieren eines mehrteiligen Uploads den Header `x-amz-server-side-encryption` hinzufügen. Beim Kopieren eines vorhandenen Objekts wird das Zielobjekt unabhängig davon, ob das Quellobjekt verschlüsselt ist, nur dann verschlüsselt, wenn Sie die serverseitige Verschlüsselung explizit anfordern.

Die Antwort-Header der folgenden REST-API-Operationen geben den Header `x-amz-server-side-encryption` zurück, wenn ein Objekt unter Verwendung von SSE-S3 gespeichert wird.

- PUT Object
- PUT Object – Kopieren
- POST Object
- Initiieren eines mehrteiligen Uploads
- Upload Part
- Hochladen eines Teiluploads – Kopieren
- Abschließen eines mehrteiligen Uploads
- Get Object
- Head Object

> **Note**
>
> Senden Sie keine Anforderungs-Header für die Verschlüsselung für `GET`- und `HEAD`-Anforderungen, wenn Ihr Objekt SSE-S3 verwendet oder der HTTP-Statuscodefehler 400 (Ungültige Anfrage) zurückgegeben wird.

Verwenden der AWS SDKs

Bei Verwendung von - AWS SDKs können Sie Amazon S3 auffordern, serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) zu verwenden. Dieser Abschnitt enthält Beispiele für die Verwendung der - AWS SDKs in mehreren Sprachen. Informationen zu anderen SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Java

Wenn Sie ein Objekt mit der AWS SDK for Java hochladen, können Sie es mit SSE-S3 verschlüsseln. Um eine serverseitige Verschlüsselung anzufordern, legen Sie mit der `ObjectMetadata`-Eigenschaft der `PutObjectRequest` den Anforderungs-Header `x-amz-server-side-encryption` fest. Wenn Sie die Methode `putObject()` des `AmazonS3Client`-Clients aufrufen, verschlüsselt und speichert Amazon S3 die Daten.

Sie können auch die Verschlüsselung mit SSE-S3 anfordern, wenn Sie Objekte mit der API-Operation für mehrteilige Uploads hochladen:

- Wenn Sie die High-Level-API-Operation für mehrteilige Uploads verwenden, wenden Sie mit der Methode `TransferManager` serverseitige Verschlüsselung auf Objekte an, während Sie sie hochladen. Sie können eine beliebige der Upload-Methoden nutzen, die `ObjectMetadata` als Parameter entgegennehmen. Weitere Informationen finden Sie unter Hochladen eines Objekts mit Multipart-Upload.
- Wenn Sie die Low-Level-API-Operation für mehrteilige Uploads verwenden, legen Sie die serverseitige Verschlüsselung beim Initiieren des mehrteiligen Uploads fest. Sie fügen die Eigenschaft `ObjectMetadata` beim Aufruf der Methode `InitiateMultipartUploadRequest.setObjectMetadata()` hinzu. Weitere Informationen finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

Sie können den Verschlüsselungsstatus eines Objekts (Verschlüsseln eines unverschlüsselten Objekts oder Entschlüsseln eines verschlüsselten Objekts) nicht direkt ändern. Um den Verschlüsselungsstatus eines Objekts zu ändern, erstellen Sie eine Kopie des Objekts, geben dabei den gewünschten Verschlüsselungsstatus der Kopie an und löschen dann das Originalobjekt. Amazon S3 verschlüsselt das kopierte Objekt nur, wenn Sie explizit eine serverseitige Verschlüsselung anfordern. Um die Verschlüsselung des kopierten Objekts über die Java-API anzufordern, geben Sie unter Verwendung der `ObjectMetadata`-Eigenschaft eine serverseitige Verschlüsselung in der `CopyObjectRequest` an.

Example Beispiel

Das folgende Beispiel veranschaulicht, wie Sie die serverseitige Verschlüsselung unter Verwendung des AWS SDK for Java festlegen. Es veranschaulicht, wie Sie die folgenden Aufgaben ausführen:

- Laden Sie ein neues Objekt mit SSE-S3 hoch.
- Ändern des Verschlüsselungsstatus eines Objekts (in diesem Beispiel Verschlüsseln eines zuvor unverschlüsselten Objekts) durch Anfertigen einer Kopie des Objekts
- Überprüfen des Verschlüsselungsstatus des Objekts

Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter Verwenden der REST-API. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.internal.SSEResultBase;
import com.amazonaws.services.s3.model.*;

import java.io.ByteArrayInputStream;

public class SpecifyServerSideEncryption {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyNameToEncrypt = "*** Key name for an object to upload and encrypt
 ***";
        String keyNameToCopyAndEncrypt = "*** Key name for an unencrypted object to
 be encrypted by copying ***";
        String copiedObjectKeyName = "*** Key name for the encrypted copy of the
 unencrypted object ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
```

```
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Upload an object and encrypt it with SSE.
            uploadObjectWithSSEEncryption(s3Client, bucketName, keyNameToEncrypt);

            // Upload a new unencrypted object, then change its encryption state
            // to encrypted by making a copy.
            changeSSEEncryptionStatusByCopying(s3Client,
                    bucketName,
                    keyNameToCopyAndEncrypt,
                    copiedObjectKeyName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void uploadObjectWithSSEEncryption(AmazonS3 s3Client, String
bucketName, String keyName) {
        String objectContent = "Test object encrypted with SSE";
        byte[] objectBytes = objectContent.getBytes();

        // Specify server-side encryption.
        ObjectMetadata objectMetadata = new ObjectMetadata();
        objectMetadata.setContentLength(objectBytes.length);

objectMetadata.setSSEAlgorithm(ObjectMetadata.AES_256_SERVER_SIDE_ENCRYPTION);
        PutObjectRequest putRequest = new PutObjectRequest(bucketName,
                keyName,
                new ByteArrayInputStream(objectBytes),
                objectMetadata);

        // Upload the object and check its encryption status.
        PutObjectResult putResult = s3Client.putObject(putRequest);
        System.out.println("Object \"" + keyName + "\" uploaded with SSE.");
        printEncryptionStatus(putResult);
    }
```

```
    private static void changeSSEEncryptionStatusByCopying(AmazonS3 s3Client,
            String bucketName,
            String sourceKey,
            String destKey) {
        // Upload a new, unencrypted object.
        PutObjectResult putResult = s3Client.putObject(bucketName, sourceKey,
 "Object example to encrypt by copying");
        System.out.println("Unencrypted object \"" + sourceKey + "\" uploaded.");
        printEncryptionStatus(putResult);

        // Make a copy of the object and use server-side encryption when storing the
        // copy.
        CopyObjectRequest request = new CopyObjectRequest(bucketName,
                sourceKey,
                bucketName,
                destKey);
        ObjectMetadata objectMetadata = new ObjectMetadata();

 objectMetadata.setSSEAlgorithm(ObjectMetadata.AES_256_SERVER_SIDE_ENCRYPTION);
        request.setNewObjectMetadata(objectMetadata);

        // Perform the copy operation and display the copy's encryption status.
        CopyObjectResult response = s3Client.copyObject(request);
        System.out.println("Object \"" + destKey + "\" uploaded with SSE.");
        printEncryptionStatus(response);

        // Delete the original, unencrypted object, leaving only the encrypted copy
 in
        // Amazon S3.
        s3Client.deleteObject(bucketName, sourceKey);
        System.out.println("Unencrypted object \"" + sourceKey + "\" deleted.");
    }

    private static void printEncryptionStatus(SSEResultBase response) {
        String encryptionStatus = response.getSSEAlgorithm();
        if (encryptionStatus == null) {
            encryptionStatus = "Not encrypted with SSE";
        }
        System.out.println("Object encryption status is: " + encryptionStatus);
    }
}
```

.NET

Wenn Sie ein Objekt hochladen, können Sie Amazon S3 dazu anweisen, es zu verschlüsseln. Um den Verschlüsselungsstatus eines vorhandenen Objekts zu ändern, erstellen Sie eine Kopie des Objekts und löschen dann das Quellobjekt. Beachten Sie, dass die Kopieroperation das Ziel nur verschlüsselt, wenn Sie auf dem Zielobjekt ausdrücklich eine serverseitige Verschlüsselung anfordern. Fügen Sie Folgendes hinzu, um SSE-S3 in `CopyObjectRequest` anzugeben:

```
ServerSideEncryptionMethod = ServerSideEncryptionMethod.AES256
```

Ein funktionierendes Beispiel, das zeigt, wie ein Objekt kopiert wird, finden Sie unter Verwenden der AWS SDKs.

Das folgende Beispiel lädt ein Objekt hoch. In der Anfrage weist das Beispiel Amazon S3 dazu an, das Objekt zu verschlüsseln. Das Beispiel ruft dann Objekt-Metadaten ab und überprüft die verwendete Verschlüsselungsmethode. Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class SpecifyServerSideEncryptionTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for object created ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            WritingAnObjectAsync().Wait();
        }

        static async Task WritingAnObjectAsync()
```

```
        {
            try
            {
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    ContentBody = "sample text",
                    ServerSideEncryptionMethod = ServerSideEncryptionMethod.AES256
                };

                var putResponse = await client.PutObjectAsync(putRequest);

                // Determine the encryption state of an object.
                GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };
                GetObjectMetadataResponse response = await
 client.GetObjectMetadataAsync(metadataRequest);
                ServerSideEncryptionMethod objectEncryption =
 response.ServerSideEncryptionMethod;

                Console.WriteLine("Encryption method used: {0}",
 objectEncryption.ToString());
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

PHP

In diesem Thema wird gezeigt, wie Sie Klassen aus Version 3 des verwenden, AWS SDK for PHP um SSE-S3 zu Objekten hinzuzufügen, die Sie in Amazon S3 hochladen. Es wird davon ausgegangen, dass Sie den Anweisungen für folgen Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen und AWS SDK for PHP ordnungsgemäß installiert ist.

Um ein Objekt zu Amazon S3 hochzuladen, verwenden Sie die Methode Aws\S3\S3Client::putObject(). Um Ihrer Upload-Anfrage den Anfrage-Header `x-amz-server-side-encryption` hinzuzufügen, geben Sie den Parameter `ServerSideEncryption` mit dem Wert `AES256` an, wie im folgenden Codebeispiel veranschaulicht. Weitere Informationen zur serverseitigen Verschlüsselungsanforderungen finden Sie unter Verwenden der REST-API.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

// $filepath should be an absolute path to a file on disk.
$filepath = '*** Your File Path ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Upload a file with server-side encryption.
$result = $s3->putObject([
    'Bucket'               => $bucket,
    'Key'                  => $keyname,
    'SourceFile'           => $filepath,
    'ServerSideEncryption' => 'AES256',
]);
```

Amazon S3 gibt als Antwort den `x-amz-server-side-encryption`-Header mit dem Wert zurück, den der Verschlüsselungsalgorithmus für die Verschlüsselung Ihrer Objektdaten verwendet hat.

Wenn Sie große Objekte mithilfe der API-Operation für mehrteilige Uploads hochladen, können Sie wie folgt SSE-S3 für diese Objekte angeben:

- Wenn Sie die Low-Level-API-Operation für mehrteilige Uploads verwenden, geben Sie die serverseitige Verschlüsselung an, wenn Sie die Methode Aws\S3\S3Client ::createMultipartUpload() aufrufen. Um Ihrer Upload-Anfrage den Anfrage-Header `x-amz-server-side-encryption` hinzuzufügen, geben Sie für den Parameter `array` den `ServerSideEncryption`-Schlüssel mit dem Wert `AES256` an. Weitere Informationen zur Low-Level-API-Operation für mehrteilige Uploads finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).
- Wenn Sie die High-Level-API-Operation für mehrteilige Uploads verwenden, geben Sie die serverseitige Verschlüsselung mithilfe des `ServerSideEncryption` Parameters der CreateMultipartUpload API-Operation an. Ein Beispiel für die Verwendung der Methode `setOption()` mit der High-Level-API-Operation für mehrteilige Uploads finden Sie unter Hochladen eines Objekts mit Multipart-Upload.

Um den Verschlüsselungsstatus eines vorhandenen Objekts zu bestimmen, rufen Sie die Objektmetadaten mit der Methode Aws\S3\S3Client::headObject() ab, wie im folgenden PHP-Codebeispiel veranschaulicht.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Check which server-side encryption algorithm is used.
$result = $s3->headObject([
    'Bucket' => $bucket,
    'Key'    => $keyname,
]);
echo $result['ServerSideEncryption'];
```

Um den Verschlüsselungsstatus eines vorhandenen Objekts zu ändern, erstellen Sie mittels der Methode Aws\S3\S3Client::copyObject() eine Kopie des Objekts und löschen dann das Quellobjekt. Standardmäßig verschlüsselt `copyObject()` das Ziel nicht, es sei denn, Sie fordern explizit die serverseitige Verschlüsselung des Zielobjekts an, indem Sie den Parameter `ServerSideEncryption` mit dem Wert `AES256` angeben. Im folgenden PHP-Codebeispiel wird eine Kopie eines Objekts erstellt und dem kopierten Objekt eine serverseitige Verschlüsselung hinzugefügt.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$sourceBucket = '*** Your Source Bucket Name ***';
$sourceKeyname = '*** Your Source Object Key ***';

$targetBucket = '*** Your Target Bucket Name ***';
$targetKeyname = '*** Your Target Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Copy an object and add server-side encryption.
$s3->copyObject([
    'Bucket'               => $targetBucket,
    'Key'                  => $targetKeyname,
    'CopySource'           => "$sourceBucket/$sourceKeyname",
    'ServerSideEncryption' => 'AES256',
]);
```

Weitere Informationen finden Sie unter den folgenden Themen:

- AWS SDK for PHP für Amazon S3 Aws\S3\S3Client-Klasse
- AWS SDK for PHP -Dokumentation:

Ruby

Wenn Sie die AWS SDK for Ruby zum Hochladen eines Objekts verwenden, können Sie angeben, dass das Objekt im Ruhezustand mit SSE-S3 verschlüsselt gespeichert werden soll. Wenn Sie das Objekt zurücklesen, wird es automatisch entschlüsselt.

Das folgende Beispiel für AWS SDK for Ruby Version 3 zeigt, wie Sie angeben, dass eine in Amazon S3 hochgeladene Datei im Ruhezustand verschlüsselt werden soll.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectPutSseWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object_encrypted(object_content, encryption)
    @object.put(body: object_content, server_side_encryption: encryption)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put your content to #{object.key}. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-encrypted-content"
  object_content = "This is my super-secret content."
  encryption = "AES256"

  wrapper = ObjectPutSseWrapper.new(Aws::S3::Object.new(bucket_name,
 object_content))
  return unless wrapper.put_object_encrypted(object_content, encryption)

  puts "Put your content into #{bucket_name}:#{object_key} and encrypted it with
 #{encryption}."
end
```

```
run_demo if $PROGRAM_NAME == __FILE__
```

Das folgende Beispiel zeigt, wie Sie den Verschlüsselungsstatus eines vorhandenen Objekts bestimmen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectGetEncryptionWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object into memory.
  #
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
 successful; otherwise nil.
  def get_object
    @object.get
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectGetEncryptionWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  obj_data = wrapper.get_object
  return unless obj_data

  encryption = obj_data.server_side_encryption.nil? ? "no" :
 obj_data.server_side_encryption
  puts "Object #{object_key} uses #{encryption} encryption."
end
```

```
run_demo if $PROGRAM_NAME == __FILE__
```

Wenn die serverseitige Verschlüsselung für das in Amazon S3 gespeicherte Objekt nicht verwendet wird, gibt die Methode `null` zurück.

Um den Verschlüsselungsstatus eines vorhandenen Objekts zu ändern, erstellen Sie eine Kopie des Objekts und löschen dann das Quellobjekt. Standardmäßig verschlüsseln die Methoden zum Kopieren das Ziel nicht, es sei denn, Sie fordern explizit die serverseitige Verschlüsselung an. Sie können die Verschlüsselung des Zielobjekts anfordern, indem Sie den Wert `server_side_encryption` im Hash-Argument der Option angeben, wie im folgenden Ruby-Codebeispiel gezeigt. Das Code-Beispiel zeigt, wie ein Objekt kopiert und die Kopie mit SSE-S3 verschlüsselt wird.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyEncryptWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                        copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket, rename it with the target
 key, and encrypt it.
  #
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key, encryption)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key,
 server_side_encryption: encryption)
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's why:
 #{e.message}"
  end
```

```
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"
  target_encryption = "AES256"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyEncryptWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key, target_encryption)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key} and "\
       "encrypted the target with #{target_object.server_side_encryption}
 encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Verwenden der AWS CLI

Verwenden Sie das folgende Beispiel, um SSE-S3 beim Hochladen eines AWS CLI Objekts mithilfe der anzugeben.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET1 --key object-key-name --server-side-
encryption AES256  --body file path
```

Weitere Informationen finden Sie unter put-object in der AWS CLI -Referenz. Informationen zur Angabe von SSE-S3 beim Kopieren eines Objekts mithilfe der AWS CLI finden Sie unter copy-object .

Verwenden von AWS CloudFormation

Beispiele für die Einrichtung der Verschlüsselung mit AWS CloudFormation finden Sie unter Erstellen eines Buckets mit Standardverschlüsselung und Erstellen eines Buckets mit AWS KMS serverseitiger Verschlüsselung mit einem S3-Bucket-Schlüssel im `AWS::S3::Bucket` `ServerSideEncryptionRule` Thema im AWS CloudFormation -Benutzerhandbuch.

Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS)

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Serverseitige Verschlüsselung ist die Verschlüsselung von Daten am Zielort durch die Anwendung oder den Service, der sie erhält.

Amazon S3 aktiviert für neue Objekt-Uploads automatisch die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

Sofern Sie nichts anderes angeben, verwenden Buckets standardmäßig SSE-S3 zum Verschlüsseln von Objekten. Sie können Buckets jedoch so konfigurieren, dass stattdessen die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) verwendet wird. Weitere Informationen finden Sie unter Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS).

AWS KMS ist ein Service, der sichere, hochverfügbare Hard- und Software kombiniert, um ein für die Cloud skaliertes Schlüsselverwaltungssystem bereitzustellen. Amazon S3 verwendet serverseitige Verschlüsselung mit AWS KMS (SSE-KMS), um Ihre S3-Objektdaten zu verschlüsseln. Wenn SSE-KMS für das Objekt angefordert wird, wird die S3-Prüfsumme (als Teil der Metadaten des Objekts) außerdem in verschlüsselter Form gespeichert. Weitere Informationen zur Prüfsumme finden Sie unter Überprüfung der Objektintegrität.

Wenn Sie KMS-Schlüssel verwenden, können Sie AWS KMS über die AWS Management Console oder die -AWS KMS API Folgendes tun:

- Sie können KMS-Schlüssel zentral erstellen, anzeigen, bearbeiten, überwachen, aktivieren oder deaktivieren, rotieren sowie das Löschen von KMS-Schlüsseln planen.
- Definieren Sie die Richtlinien, die steuern, wie und von wem KMS-Schlüssel verwendet werden können.
- Prüfen Sie deren Verwendung, um zu beweisen, dass sie korrekt verwendet werden. Das Auditing wird von der AWS KMS -API unterstützt, nicht jedoch von der AWS KMSAWS Management Console.

Die Sicherheitskontrollen in AWS KMS können Ihnen helfen, Compliance-Anforderungen im Zusammenhang mit der Verschlüsselung zu erfüllen. Sie können mit diesen KMS-Schlüsseln Ihre Daten in Amazon-S3-Buckets schützen. Wenn Sie die SSE-KMS-Verschlüsselung mit einem S3-Bucket verwenden, AWS KMS keys muss sich der in derselben Region wie der Bucket befinden.

Für die Nutzung von fallen zusätzliche Gebühren an AWS KMS keys. Weitere Informationen finden Sie unter AWS KMS key -Konzepte im AWS Key Management Service -Entwicklerhandbuch und in den AWS KMS -Preisen.

Berechtigungen

Um ein mit einem verschlüsseltes Objekt AWS KMS key in Amazon S3 hochzuladen, benötigen Sie `kms:GenerateDataKey` Berechtigungen für den Schlüssel. Um ein mit einem verschlüsseltes Objekt herunterzuladen AWS KMS key, benötigen Sie -`kms:Decrypt`Berechtigungen. Informationen zu den AWS KMS Berechtigungen, die für mehrteilige Uploads erforderlich sind, finden Sie unter API für mehrteilige Uploads und Berechtigungen.

Themen

- AWS KMS keys
- Amazon-S3-Bucket-Schlüssel
- Erzwingen serverseitiger Verschlüsselung
- Verschlüsselungskontext
- Senden von Anfragen für AWS KMS verschlüsselte Objekte
- Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS)
- Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln

AWS KMS keys

Wenn Sie die serverseitige Verschlüsselung mit AWS KMS (SSE-KMS) verwenden, können Sie den AWS standardmäßigen -verwalteten Schlüssel verwenden oder einen kundenverwalteten Schlüssel angeben, den Sie bereits erstellt haben. AWS KMS unterstützt Envelope-Verschlüsselung. S3 verwendet die AWS KMS Funktionen für die Envelope-Verschlüsselung, um Ihre Daten weiter zu schützen. Die Umschlagverschlüsselung bezeichnet das Verschlüsseln Ihrer Klartextdaten mit einem Datenschlüssel und die anschließende Verschlüsselung dieses Datenschlüssels mit einem KMS-Schlüssel. Weitere Informationen zur Envelope-Verschlüsselung finden Sie unter Envelope-Verschlüsselung im AWS Key Management Service -Entwicklerhandbuch.

Wenn Sie keinen kundenverwalteten Schlüssel angeben, erstellt Amazon S3 automatisch ein Von AWS verwalteter Schlüssel in Ihrem , AWS-Konto wenn Sie zum ersten Mal ein mit SSE-KMS verschlüsseltes Objekt zu einem Bucket hinzufügen. Standardmäßig verwendet Amazon S3 diesen KMS-Schlüssel für SSE-KMS.

> **Note**
>
> Objekte, die mit SSE-KMS mit Von AWS verwaltete Schlüssel verschlüsselt wurden, können nicht kontoübergreifend gemeinsam genutzt werden. Wenn Sie SSE-KMS-Daten kontoübergreifend freigeben müssen, müssen Sie einen vom Kunden verwalteten Schlüssel von verwenden AWS KMS.

Wenn Sie einen vom Kunden verwalteten Schlüssel für SSE-KMS verwenden möchten, können Sie einen symmetrischen kundenverwalteten Verschlüsselungsschlüssel erstellen, bevor Sie SSE-KMS konfigurieren. Wenn Sie dann SSE-KMS für Ihren Bucket konfigurieren, können Sie den vorhandenen vom Kunden verwalteten Schlüssel angeben. Weitere Informationen um symmetrischen Verschlüsselungsschlüssel finden Sie unter Symmetrische KMS-Verschlüsselungsschlüssel im Entwicklerhandbuch für AWS Key Management Service .

Durch die Erstellung eines vom Kunden verwalteten Schlüssels erhalten Sie mehr Flexibilität und Kontrolle. Beispielsweise können Sie kundenverwaltete Schlüssel erstellen, drehen und deaktivieren. Sie können auch Zugriffskontrollen definieren und die vom Kunden verwalteten Schlüssel prüfen, mit denen Sie Ihre Daten schützen. Weitere Informationen zu vom Kunden verwalteten und AWS von verwalteten Schlüsseln finden Sie unter Kundenschlüssel und AWS -Schlüssel im AWS Key Management Service Entwicklerhandbuch für .

> **Note**
>
> Wenn Sie serverseitige Verschlüsselung mit einem kundenverwalteten Schlüssel verwenden, der in einem externen Schlüsselspeicher abgelegt ist, sind Sie im Gegensatz zu Standard-KMS-Schlüsseln dafür verantwortlich, die Verfügbarkeit und Beständigkeit Ihres Schlüsselmaterials sicherzustellen. Weitere Informationen zu externen Schlüsselspeichern und dazu, wie sie das Modell der geteilten Verantwortung verändern, finden Sie unter Externe Schlüsselspeicher im Entwicklerhandbuch für AWS Key Management Service .

Wenn Sie Ihre Daten mit einem Von AWS verwalteter Schlüssel oder einem vom Kunden verwalteten Schlüssel verschlüsseln möchten, AWS KMS führen und Amazon S3 die folgenden Envelope-Verschlüsselungsaktionen aus:

1. Amazon S3 fordert einen Klartext-Datenschlüssel und eine mit dem angegebenen KMS-Schlüssel verschlüsselte Kopie des Schlüssels an.
2. AWS KMS generiert einen Datenschlüssel, verschlüsselt ihn mit dem KMS-Schlüssel und sendet sowohl den Klartext-Datenschlüssel als auch den verschlüsselten Datenschlüssel an Amazon S3.
3. Amazon S3 verschlüsselt die Daten mit dem Datenschlüssel und entfernt anschließend den Klartextschlüssel schnellstmöglich aus dem Arbeitsspeicher.
4. Amazon S3 speichert den verschlüsselten Datenschlüssel im Metadatenformat zusammen mit den verschlüsselten Daten.

Wenn Sie anfordern, dass Ihre Daten entschlüsselt werden, AWS KMS führen Amazon S3 und die folgenden Aktionen aus:

1. Amazon S3 sendet den verschlüsselten Datenschlüssel an AWS KMS in einer -`Decrypt`Anforderung.
2. AWS KMS entschlüsselt den verschlüsselten Datenschlüssel mit demselben KMS-Schlüssel und gibt den Klartext-Datenschlüssel an Amazon S3 zurück.
3. Amazon S3 entschlüsselte die verschlüsselten Daten mit dem Klartext-Datenschlüssel und entfernt den Klartext-Datenschlüssel anschließend schnellstmöglich aus dem Arbeitsspeicher.

> **Important**
>
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung. Weitere Informationen zu diesen Schlüsseln finden Sie unter Symmetrische KMS-Verschlüsselungsschlüssel im Entwicklerhandbuch für AWS Key Management Service .

Wenn Sie Anfragen identifizieren möchten, die SSE-KMS angeben, können Sie die Metriken Alle SSE-KMS-Anfragen und % aller SSE-KMS-Anfragen der Metriken von Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Weitere Informationen finden Sie unter Bewertung Ihrer Speicheraktivität und -nutzung mit S3 Storage Lens. Eine vollständige Liste der Metriken finden Sie im Glossar der S3-Storage-Lens-Metriken.

Amazon-S3-Bucket-Schlüssel

Wenn Sie die serverseitige Verschlüsselung mit AWS KMS (SSE-KMS) konfigurieren, können Sie Ihre Buckets für die Verwendung von S3-Bucket-Schlüsseln für SSE-KMS konfigurieren. Die Verwendung eines Schlüssels auf Bucket-Ebene für SSE-KMS kann Ihre AWS KMS Anforderungskosten um bis zu 99 Prozent senken, indem der Anforderungsdatenverkehr von Amazon S3 zu verringert wird AWS KMS.

Wenn Sie einen Bucket für die Verwendung von S3-Bucket-Schlüsseln für SSE-KMS bei neuen Objekten konfigurieren, generiert AWS KMS einen Schlüssel auf Bucket-Ebene, mit dem eindeutige Datenschlüssel für Objekte im Bucket erstellt werden. Dieser S3-Bucket-Schlüssel wird für einen zeitlich begrenzten Zeitraum in Amazon S3 verwendet, wodurch Amazon S3 weitere Anforderungen an stellen muss AWS KMS , um Verschlüsselungsvorgänge abzuschließen. Weitere Informationen zur Verwendung von S3-Bucket-Schlüsseln finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

Erzwingen serverseitiger Verschlüsselung

Wenn Sie die serverseitige Verschlüsselung aller Objekte in einem bestimmten Amazon-S3-Bucket anfordern möchten, können Sie eine Bucket-Richtlinie verwenden. Beispielsweise verweigert die folgende Bucket-Richtlinie jedem die Berechtigung zum Hochladen von Objekten (`s3:PutObject`), wenn die Anforderung nicht den Header `x-amz-server-side-encryption-aws-kms-key-id` enthält, der die serverseitige Verschlüsselung mit SSE-KMS anfordert.

```
{
   "Version":"2012-10-17",
   "Id":"PutObjectPolicy",
   "Statement":[{
         "Sid":"DenyObjectsThatAreNotSSEKMS",
         "Effect":"Deny",
         "Principal":"*",
         "Action":"s3:PutObject",
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
         "Condition":{
            "Null":{
               "s3:x-amz-server-side-encryption-aws-kms-key-id":"true"
            }
         }
      }
   ]
}
```

Um zu verlangen, dass ein bestimmter zum Verschlüsseln der Objekte in einem Bucket verwendet AWS KMS key wird, können Sie den `s3:x-amz-server-side-encryption-aws-kms-key-id` Bedingungsschlüssel verwenden. Um den KMS-Schlüssel anzugeben, müssen Sie einen Amazon-Ressourcennamen (ARN) des AWS Identity and Access Management Schlüssels im `arn:aws:kms:`*`region`*`:`*`acct-id`*`:key/`*`key-id`* Format verwenden. überprüft nicht, ob die Zeichenfolge für `s3:x-amz-server-side-encryption-aws-kms-key-id` vorhanden ist.

> **Note**
>
> Wenn Sie ein Objekt hochladen, können Sie den KMS-Schlüssel über den Header `x-amz-server-side-encryption-aws-kms-key-id` festlegen. Wenn der Header in der Anforderung nicht vorhanden ist, geht Amazon S3 davon aus, dass Sie den Von AWS verwalteter Schlüssel verwenden möchten. Unabhängig davon muss die AWS KMS Schlüssel-ID, die Amazon S3 für die Objektverschlüsselung verwendet, mit der AWS KMS Schlüssel-ID in der Richtlinie übereinstimmen, andernfalls lehnt Amazon S3 die Anforderung ab.

Eine vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Verschlüsselungskontext

Ein Verschlüsselungskontext ist ein Satz von Schlüssel-Wert-Paaren, die zusätzliche kontextbezogene Informationen zu den Daten enthalten können. Der Verschlüsselungskontext ist nicht verschlüsselt. Wenn für eine Verschlüsselungsoperation ein Verschlüsselungskontext angegeben wird, muss Amazon S3 denselben Verschlüsselungskontext auch für die Entschlüsselungsoperation angeben. Andernfalls schlägt die Entschlüsselung fehl. AWS KMS verwendet den Verschlüsselungskontext als zusätzliche authentifizierte Daten (AAD), um die authentifizierte Verschlüsselung zu unterstützen. Weitere Informationen zum Verschlüsselungs-Kontext finden Sie unter Encryption Context (Verschlüsselungs-Kontext) im AWS Key Management Service -Entwicklerhandbuch.

Amazon S3 verwendet automatisch den Amazon-Ressourcennamen (ARN) des Objekts oder Buckets als Verschlüsselungskontextpaar:

- Wenn Sie SSE-KMS verwenden, ohne einen S3-Bucket-Schlüssel zu aktivieren, verwenden Sie den Objekt-ARN als Verschlüsselungskontext.

  ```
  arn:aws:s3:::object_ARN
  ```

- Wenn Sie SSE-KMS verwenden und einen S3-Bucket-Schlüssel aktivieren, wird der Bucket-ARN als Verschlüsselungskontext verwendet. Weitere Informationen zu S3-Bucket-Schlüsseln finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

  ```
  arn:aws:s3:::bucket_ARN
  ```

Sie können optional ein zusätzliches Verschlüsselungskontextpaar bereitstellen, indem Sie den `-x-amz-server-side-encryption-context`Header in einer s3:PutObject-Anforderung verwenden. Da der Verschlüsselungskontext jedoch nicht verschlüsselt ist, sollte er keine sensiblen Informationen enthalten. Amazon S3 speichert dieses zusätzliche Schlüsselpaar zusammen mit dem Standardverschlüsselungskontext. Wenn Ihre PUT-Anforderung verarbeitet wird, hängt Amazon S3 den Standardverschlüsselungskontext `aws:s3:arn` an den von Ihnen bereitgestellten an.

Sie können den Verschlüsselungskontext verwenden, um Ihre kryptografischen Vorgänge zu identifizieren und zu kategorisieren. Sie können auch den ARN-Wert des Standardverschlüsselungskontexts verwenden, um relevante Anfragen in zu verfolgen, AWS CloudTrail indem Sie anzeigen, welcher Amazon S3-ARN mit welchem Verschlüsselungsschlüssel verwendet wurde.

Im `requestParameters` Feld einer CloudTrail Protokolldatei sieht der Verschlüsselungskontext ähnlich wie der folgende aus.

```
"encryptionContext": {
    "aws:s3:arn": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/file_name"
}
```

Wenn Sie SSE-KMS mit der optionalen S3-Bucket-Keys-Funktion verwenden, ist der Verschlüsselungskontextwert der ARN des Buckets.

```
"encryptionContext": {
    "aws:s3:arn": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
}
```

Senden von Anfragen für AWS KMS verschlüsselte Objekte

- **Important**

  Alle - `GET` und -`PUT`Anfragen für AWS KMS verschlüsselte Objekte müssen mit Secure Sockets Layer (SSL) oder Transport Layer Security (TLS) gestellt werden. Anforderungen müssen auch mit gültigen Anmeldeinformationen signiert werden, z. B. mit AWS Signature Version 4 (oder AWS Signature Version 2).

  AWS Signature Version 4 ist der Prozess zum Hinzufügen von Authentifizierungsinformationen zu AWS Anforderungen, die über HTTP gesendet werden. Aus Sicherheitsgründen AWS müssen die meisten Anfragen an mit einem Zugriffsschlüssel signiert werden, der aus einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel besteht. Diese beiden Schlüssel werden in der Regel als Sicherheitsanmeldeinformationen bezeichnet. Weitere Informationen finden Sie unter Authenticating Requests (Authentifizierung von Anforderungen) (AWS Signature Version 4) und Signature Version 4 signing process (Signaturprozess).

- **Important**

  Wenn Ihr Objekt SSE-KMS verwendet, dürfen Sie keine Verschlüsselungsanforderungs-Header für `GET`- und `HEAD`-Anforderungen senden. Andernfalls erhalten Sie den Fehler HTTP 400 Bad Request.

Themen

Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS)

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert und alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden im Ruhezustand automatisch verschlüsselt. Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) ist die Standardverschlüsselungskonfiguration für jeden Bucket in Amazon S3. Um einen anderen Verschlüsselungstyp zu verwenden, können Sie entweder die Art der serverseitigen Verschlüsselung angeben, die in Ihren S3-PUT-Anfragen verwendet werden soll, oder Sie können die Standardverschlüsselungskonfiguration im Ziel-Bucket festlegen.

Wenn Sie in Ihren PUT Anforderungen einen anderen Verschlüsselungstyp angeben möchten, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) oder die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden. Wenn Sie im Ziel-Bucket eine andere Standardverschlüsselungskonfiguration festlegen möchten, können Sie SSE-KMS oder DSSE-KMS verwenden.

Sie können die Verschlüsselung anwenden, wenn Sie entweder ein neues Objekt hochladen oder ein vorhandenes Objekt kopieren.

Sie können SSE-KMS mithilfe der Amazon S3-Konsole, REST-API-Operationen, AWS SDKs und der AWS Command Line Interface (AWS CLI) angeben. Weitere Informationen finden Sie unter den folgenden Themen.

> **Note**
>
> Sie können Multi-Region AWS KMS keys in Amazon S3 verwenden. Amazon S3 behandelt jedoch derzeit Multi-Regions-Schlüssel wie Einzel-Regions-Schlüssel und verwendet nicht die Multi-Regions-Funktionen des Schlüssels. Weitere Informationen finden Sie unter Using multi-Region keys (Verwenden von Multi-Regions-Zugriffpunkt-Schlüsseln) im AWS Key Management Service -Entwicklerhandbuch.

> **Note**
>
> Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen. Weitere Informationen zu kontoübergreifenden Berechtigungen für KMS-Schlüssel finden Sie unter Erstellen von KMS-Schlüsseln, die von anderen Konten verwendet werden können im Entwicklerhandbuch zu AWS Key Management Service .

Verwenden der S3-Konsole

In diesem Thema wird beschrieben, wie Sie den Verschlüsselungstyp eines Objekts festlegen oder ändern, um die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) mithilfe der Amazon S3-Konsole zu verwenden.

> **Note**
>
> Wenn Sie die Verschlüsselung eines Objekts ändern, wird ein neues Objekt erstellt, um das alte zu ersetzen. Wenn S3-Versioning aktiviert ist, wird eine neue Version des Objekts erstellt, und das vorhandene Objekt wird zu einer älteren Version. Die Rolle, die die Eigenschaft ändert, wird auch Besitzer des neuen Objekts (oder der neuen Objektversion).

So fügen Sie die Verschlüsselung für ein Objekt hinzu oder ändern Sie sie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
4. Wählen Sie in der Liste Objekte den Namen des Objekts aus, für das Sie eine Verschlüsselung hinzufügen oder ändern möchten.

   Die Detailseite des Objekts wird angezeigt. Sie enthält mehrere Abschnitte mit den Eigenschaften des Objekts.
5. Wählen Sie die Registerkarte Eigenschaften aus.
6. Scrollen Sie nach unten zum Abschnitt Serverseitige Verschlüsselungseinstellungen und wählen Sie Bearbeiten aus.

   Die Seite Serverseitige Verschlüsselung bearbeiten wird geöffnet.
7. Wählen Sie unter Serverseitige Verschlüsselung für Verschlüsselungseinstellungen die Option Überschreiben der Standardverschlüsselungs-Bucket-Einstellungen aus.
8. Wählen Sie unter Verschlüsselungstyp die Option Serverseitige Verschlüsselung mit - AWS Key Management Service Schlüsseln (SSE-KMS) aus.

   > **Important**
   >
   > Wenn Sie die Option SSE-KMS für die Standardverschlüsselung verwenden, unterliegen Sie den Kontingenten der Anforderungen pro Sekunde (RPS) von AWS KMS. Weitere Informationen zu AWS KMS -Kontingenten und zum Anfordern einer Kontingenterhöhung finden Sie unter Kontingente im Entwicklerhandbuch zu AWS Key Management Service .
9. Führen Sie unter AWS KMS -Schlüssel eine der folgenden Aktionen aus, um Ihren KMS-Schlüssel auszuwählen:

   - Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS keys wählen und dann den KMS-Schlüssel in der Liste der verfügbaren Schlüssel aus.

     Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom

Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS -Schlüssel im Entwicklerhandbuch zu AWS Key Management Service .

- Um den KMS-Schlüssel-ARN einzugeben, wählen Sie AWS KMS key ARN eingeben und geben Sie dann Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
- Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

  Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

> **Important**
>
> Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket verfügbar sind. Die Amazon-S3-Konsole führt nur die ersten 100 KMS-Schlüssel auf, die in derselben Region wie der Bucket verfügbar sind. Wenn Sie einen KMS-Schlüssel verwenden möchten, der nicht aufgeführt ist, müssen Sie den KMS-Schlüssel-ARN eingeben. Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen und Sie müssen den KMS-Schlüssel-ARN eingeben.
> Amazon S3 unterstützt nur symmetrisch verschlüsselte KMS-Schlüssel und keine asymmetrischen KMS-Schlüssel. Weitere Informationen finden Sie unter Erkennen von symmetrischen und asymmetrischen KMS-Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service .

10. Wählen Sie Save Changes (Änderungen speichern).

> **Note**
>
> Diese Aktion wendet auf alle angegebenen Objekte Verschlüsselung an. Warten Sie beim Verschlüsseln von Ordnern, bis die Speicheroperation abgeschlossen ist, bevor Sie dem Ordner neue Objekte hinzufügen.

Verwenden der REST-API

Wenn Sie ein Objekt erstellen, d. h. wenn Sie ein neues Objekt hochladen oder ein vorhandenes Objekt kopieren, können Sie für die Verschlüsselung Ihrer Daten die serverseitige Verschlüsselung mit AWS KMS keys (SSE-KMS) angeben. Fügen Sie hierzu der Anforderung den Header `x-amz-server-side-encryption` hinzu. Setzen Sie den Wert des Headers auf den `aws:kms`-Verschlüsselungsalgorithmus. Amazon S3 bestätigt, dass Ihr Objekt unter Verwendung von SSE-KMS gespeichert wird, indem es den Antwort-Header `x-amz-server-side-encryption` zurückgibt.

Wenn Sie den Header `x-amz-server-side-encryption` mit dem Wert `aws:kms` angeben, können Sie auch die folgenden Anforderungs-Header verwenden:

- `x-amz-server-side-encryption-aws-kms-key-id`
- `x-amz-server-side-encryption-context`
- `x-amz-server-side-encryption-bucket-key-enabled`

Themen

Amazon-S3-REST-API-Vorgänge, die SSE-KMS unterstützen

Die folgenden REST-API-Vorgänge akzeptieren die Anforderungs-Header `x-amz-server-side-encryption`, `x-amz-server-side-encryption-aws-kms-key-id` und `x-amz-server-side-encryption-context`.

- PutObject – Wenn Sie Daten mithilfe der PUT -API-Operation hochladen, können Sie diese Anforderungs-Header angeben.
- CopyObject – Wenn Sie ein Objekt kopieren, haben Sie sowohl ein Quellobjekt als auch ein Zielobjekt. Wenn Sie SSE-KMS-Header mit der `CopyObject`-Operation übergeben, werden sie nur auf das Zielobjekt angewendet. Beim Kopieren eines vorhandenen Objekts wird das Zielobjekt unabhängig davon, ob das Quellobjekt verschlüsselt ist, nur dann verschlüsselt, wenn Sie die serverseitige Verschlüsselung explizit anfordern.

- POST Object – Wenn Sie eine POST-Operation für das Hochladen eines Objekts verwenden, geben Sie die Informationen in die Formularfelder und nicht in die Anforderungs-Header ein.
- CreateMultipartUpload – Wenn Sie große Objekte mithilfe der API-Operation für mehrteilige Uploads hochladen, können Sie diese Header angeben. Sie geben diese Header in der Anforderung zum Initiieren eines mehrteiligen Uploads an.

Die Antwort-Header der folgenden REST-API-Operationen geben den Header `x-amz-server-side-encryption` zurück, wenn ein Objekt unter Verwendung der serverseitigen Verschlüsselung gespeichert wird.

- PutObject
- CopyObject
- POST Object
- CreateMultipartUpload
- UploadPart
- UploadPartCopy
- CompleteMultipartUpload
- GetObject
- HeadObject

> **Important**
>
> - Alle GET- und PUT-Anforderungen für ein Objekt, das durch AWS KMS geschützt wird, schlagen fehl, wenn Sie diese nicht mit Secure Sockets Layer (SSL), Transport Layer Security (TLS) oder Signature Version 4 erstellen.
> - Wenn Ihr Objekt SSE-KMS verwendet, senden Sie keine Verschlüsselungsanforderungs-Header für GET -Anforderungen und -HEADAnforderungen. Andernfalls erhalten Sie einen HTTP-400 BadRequest-Fehler.

Verschlüsselungskontext (x-amz-server-side-encryption-context)

Wenn Sie `x-amz-server-side-encryption:aws:kms` angeben, unterstützt die Amazon-S3-API einen Verschlüsselungskontext mit dem Header `x-amz-server-side-encryption-`

`context`. Ein Verschlüsselungskontext ist ein Satz von Schlüssel-Wert-Paaren, die zusätzliche kontextbezogene Informationen zu den Daten enthalten können.

Amazon S3 verwendet automatisch den Objekt- oder Bucket-ARN (Amazon-Ressourcenname) als Verschlüsselungskontextpaar. Wenn Sie SSE-KMS verwenden, ohne einen S3-Bucket-Schlüssel zu aktivieren, verwenden Sie den Objekt-ARN als Verschlüsselungskontext, z. B. `arn:aws:s3:::object_ARN`. Wenn Sie dagegen SSE-KMS verwenden und einen S3-Bucket-Schlüssel aktivieren, verwenden Sie den Bucket-ARN für Ihren Verschlüsselungskontext, z. B. `arn:aws:s3:::bucket_ARN`.

Sie können optional ein zusätzliches Verschlüsselungskontextpaar bereitstellen, indem Sie den Header `x-amz-server-side-encryption-context` verwenden. Da der Verschlüsselungskontext jedoch nicht verschlüsselt ist, sollte er keine sensiblen Informationen enthalten. Amazon S3 speichert dieses zusätzliche Schlüsselpaar zusammen mit dem Standardverschlüsselungskontext.

Weitere Informationen zum Verschlüsselungskontext in Amazon S3 finden Sie unter Verschlüsselungskontext. Allgemeine Informationen zum Verschlüsselungs-Kontext finden Sie unter AWS Key Management Service Concepts – Encryption Context (Konzepte – Verschlüsselungs-Kontext) im AWS Key Management Service -Entwicklerhandbuch.

AWS KMS -Schlüssel-ID (x-amz-server-side-encryption-aws-kms-key-id)

Sie können den Header `x-amz-server-side-encryption-aws-kms-key-id` verwenden, um die ID des vom Kunden verwalteten Schlüssels anzugeben, der zum Schutz der Daten verwendet wird. Wenn Sie den Header `x-amz-server-side-encryption:aws:kms`, jedoch nicht den Header `x-amz-server-side-encryption-aws-kms-key-id` angeben, verwendet Amazon S3 Von AWS verwalteter Schlüssel (`aws/s3`), um die Daten zu schützen. Wenn Sie einen vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie den `x-amz-server-side-encryption-aws-kms-key-id`-Header des vom Kunden verwalteten Schlüssels angeben.

> **Important**
>
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung. Weitere Informationen zu diesen Schlüsseln finden Sie unter Symmetrische KMS-Verschlüsselungsschlüssel im Entwicklerhandbuch für AWS Key Management Service .

S3-Bucket-Schlüssel (x-amz-server-side-encryption-aws-bucket-key-aktiviert)

Sie können den Anforderungs-Header `x-amz-server-side-encryption-aws-bucket-key-enabled` verwenden, um einen S3-Bucket-Schlüssel auf Objektebene zu aktivieren oder zu deaktivieren. S3-Bucket-Schlüssel reduzieren Ihre AWS KMS Anforderungskosten, indem der Anforderungsdatenverkehr von Amazon S3 zu verringert wird AWS KMS. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

Wenn Sie den Header `x-amz-server-side-encryption:aws:kms`, jedoch nicht den Header `x-amz-server-side-encryption-aws-bucket-key-enabled` angeben, werden die Einstellung des S3-Bucket-Schlüssels für den Ziel-Bucket verwendet, um Ihr Objekt zu verschlüsseln. Weitere Informationen finden Sie unter Konfigurieren eines S3-Bucket-Schlüssels auf Objektebene .

Verwenden der AWS CLI

Wenn Sie ein neues Objekt hochladen oder ein vorhandenes Objekt kopieren, können Sie die Verwendung der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln zur Verschlüsselung Ihrer Daten angeben. Fügen Sie hierzu der Anforderung den Header `--server-side-encryption aws:kms` hinzu. Verwenden Sie die `--ssekms-key-id` *`example-key-id`*, um Ihren vom Kunden verwalteten AWS KMS Schlüssel hinzuzufügen, den Sie erstellt haben. Wenn Sie angeben`--server-side-encryption aws:kms`, aber keine - AWS KMS Schlüssel-ID angeben, verwendet Amazon S3 einen - AWS verwalteten Schlüssel.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key example-object-key --server-side-encryption aws:kms --ssekms-key-id example-key-id --body filepath
```

Sie können Schlüssel von Amazon-S3-Buckets zusätzlich für Ihre PUT- oder COPY-Operationen aktivieren oder deaktivieren, indem Sie `--bucket-key-enabled` oder `--no-bucket-key-enabled` hinzufügen. Amazon-S3-Bucket-Schlüssel können Ihre AWS KMS Anforderungskosten senken, indem der Anforderungsdatenverkehr von Amazon S3 zu verringert wird AWS KMS. Weitere Informationen finden Sie unter Reduzierung der Kosten für SSE-KMS mit Schlüsseln vn Amazon-S3-Buckets.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key example-object-key --server-side-encryption aws:kms --bucket-key-enabled --body filepath
```

Sie können ein unverschlüsseltes Objekt mit SSE-KMS verschlüsseln, indem Sie das Objekt wieder an seinen Platz kopieren.

```
aws s3api copy-object --bucket DOC-EXAMPLE-BUCKET --key example-object-key --
body filepath --bucket DOC-EXAMPLE-BUCKET --key example-object-key --sse aws:kms --sse-
kms-key-id example-key-id --body filepath
```

Verwenden der AWS SDKs

Wenn Sie AWS SDKs verwenden, können Sie Amazon S3 auffordern, AWS KMS keys für die serverseitige Verschlüsselung zu verwenden. Dieser Abschnitt enthält Beispiele für die Verwendung der - AWS SDKs für Java und .NET. Informationen zu anderen SDKs finden Sie unter Beispiel-Code und Bibliotheken.

> **Important**
>
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung. Weitere Informationen zu diesen Schlüsseln finden Sie unter Symmetrische KMS-Verschlüsselungsschlüssel im Entwicklerhandbuch für AWS Key Management Service .

Copy-Vorgang

Wenn Sie Objekte kopieren, fügen Sie dieselben Anfrageeigenschaften (`ServerSideEncryptionMethod` und `ServerSideEncryptionKeyManagementServiceKeyId`) hinzu, um Amazon S3 aufzufordern, ein AWS KMS key zu verwenden. Weitere Informationen über das Kopieren von Objekten finden Sie unter Objekte kopieren.

PUT-Operation

Java

Wenn Sie ein Objekt über die hochladen AWS SDK for Java, können Sie Amazon S3 auffordern, einen zu verwenden, AWS KMS key indem Sie die `SSEAwsKeyManagementParams` Eigenschaft hinzufügen, wie in der folgenden Anforderung gezeigt.

```
PutObjectRequest putRequest = new PutObjectRequest(bucketName,
   keyName, file).withSSEAwsKeyManagementParams(new SSEAwsKeyManagementParams());
```

In diesem Fall verwendet Amazon S3 die Von AWS verwalteter Schlüssel (`aws/s3`) (siehe Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS). Sie können optional einen symmetrischen KMS-Verschlüsselungsschlüssel erstellen und diesen in der Anfrage angeben.

```
PutObjectRequest putRequest = new PutObjectRequest(bucketName,
   keyName, file).withSSEAwsKeyManagementParams(new
 SSEAwsKeyManagementParams(keyID));
```

Weitere Informationen zum Erstellen von kundenverwalteten Schlüsseln finden Sie unter Programmieren der AWS KMS -API im AWS Key Management Service -Entwicklerhandbuch.

Funktionierende Codebeispiele zum Hochladen eines Objekts finden Sie unter den folgenden Themen. Um diese Beispiele zu verwenden, müssen Sie die Codebeispiele aktualisieren und Verschlüsselungsinformationen wie im vorigen Codefragment gezeigt bereitstellen.

- Weitere Informationen zum Hochladen eines Objekts in einem einzigen Vorgang finden Sie unter Objekte hochladen.
- Weitere Informationen zu mehrteiligen Uploads finden Sie unter den folgenden Themen:
  - Weitere Informationen über die Verwendung der High-Level-API für mehrteilige Uploads finden Sie unter Hochladen eines Objekts mit Multipart-Upload.
  - Weitere Informationen über die Verwendung der Low-Level-API für mehrteilige Uploads finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

.NET

Wenn Sie ein Objekt über die hochladen AWS SDK for .NET, können Sie Amazon S3 auffordern, einen zu verwenden, AWS KMS key indem Sie die `ServerSideEncryptionMethod` Eigenschaft hinzufügen, wie in der folgenden Anforderung gezeigt.

```
PutObjectRequest putRequest = new PutObjectRequest
 {
     BucketName = DOC-EXAMPLE-BUCKET,
     Key = keyName,
     // other properties.
     ServerSideEncryptionMethod = ServerSideEncryptionMethod.AWSKMS
 };
```

In diesem Fall verwendet Amazon S3 die Von AWS verwalteter Schlüssel. Weitere Informationen finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS). Sie können optional Ihren eigenen kundenverwalteten symmetrischen Verschlüsselungsschlüssel erstellen und diesen in der Anfrage angeben.

```
PutObjectRequest putRequest1 = new PutObjectRequest
{
  BucketName = DOC-EXAMPLE-BUCKET,
  Key = keyName,
  // other properties.
  ServerSideEncryptionMethod = ServerSideEncryptionMethod.AWSKMS,
  ServerSideEncryptionKeyManagementServiceKeyId = keyId
};
```

Weitere Informationen zum Erstellen von kundenverwalteten Schlüsseln finden Sie unter Programmieren der AWS KMS API im AWS Key Management Service -Entwicklerhandbuch.

Funktionierende Codebeispiele zum Hochladen eines Objekts finden Sie unter den folgenden Themen. Um diese Beispiele zu verwenden, müssen Sie die Codebeispiele aktualisieren und Verschlüsselungsinformationen wie im vorigen Codefragment gezeigt bereitstellen.

- Weitere Informationen zum Hochladen eines Objekts in einem einzigen Vorgang finden Sie unter Objekte hochladen.
- Weitere Informationen zu mehrteiligen Uploads finden Sie unter den folgenden Themen:
  - Weitere Informationen über die Verwendung der High-Level-API für mehrteilige Uploads finden Sie unter Hochladen eines Objekts mit Multipart-Upload.
  - Weitere Informationen über die Verwendung der Low-Level-API für mehrteilige Uploads finden Sie unter Hochladen eines Objekts mit Multipart-Upload.

Vorsignierte URLs

Java

Wenn Sie eine vorsignierte URL für ein mit einem verschlüsseltes Objekt erstellen AWS KMS key, müssen Sie Signature Version 4 explizit angeben.

```
ClientConfiguration clientConfiguration = new ClientConfiguration();
clientConfiguration.setSignerOverride("AWSS3V4SignerType");
AmazonS3Client s3client = new AmazonS3Client(
```

```
        new ProfileCredentialsProvider(), clientConfiguration);
...
```

Ein Codebeispiel finden Sie unter Gemeinsame Nutzung von Objekten mit vorsignierten URLs.

.NET

Wenn Sie eine vorsignierte URL für ein mit einem verschlüsseltes Objekt erstellen AWS KMS key, müssen Sie Signature Version 4 explizit angeben.

```
AWSConfigs.S3Config.UseSignatureVersion4 = true;
```

Ein Codebeispiel finden Sie unter Gemeinsame Nutzung von Objekten mit vorsignierten URLs.

Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln

Amazon S3-Bucket-Schlüssel reduzieren die Kosten für serverseitige Amazon-SAmazon S3-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS). Die Verwendung eines Schlüssels auf Bucket-Ebene für SSE-KMS kann die AWS KMS Anforderungskosten um bis zu 99 Prozent reduzieren, indem der Anforderungsdatenverkehr von Amazon S3 zu verringert wird AWS KMS. Mit ein paar Klicks in der AWS Management Console und ohne Änderungen an Ihren Client-Anwendungen können Sie Ihren Bucket so konfigurieren, dass ein S3-Bucket-Schlüssel für die SSE-KMS-Verschlüsselung bei neuen Objekten verwendet wird.

> **Note**
>
> S3-Bucket-Schlüssel werden für serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS) nicht unterstützt.

S3-Bucket-Schlüssel für SSE-KMS

Workloads, die auf Millionen oder Milliarden von Objekten zugreifen, die mit SSE-KMS verschlüsselt sind, können große Mengen von Anfragen an generieren AWS KMS. Wenn Sie SSE-KMS verwenden, um Ihre Daten ohne S3-Bucket-Schlüssel zu schützen, verwendet Amazon S3 einen individuellen AWS KMS Datenschlüssel für jedes Objekt. In diesem Fall ruft Amazon S3 AWS KMS jedes Mal auf, wenn eine Anforderung für ein KMS-verschlüsseltes Objekt gestellt wird. Informationen zur Funktionsweise von SSE-KMS finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

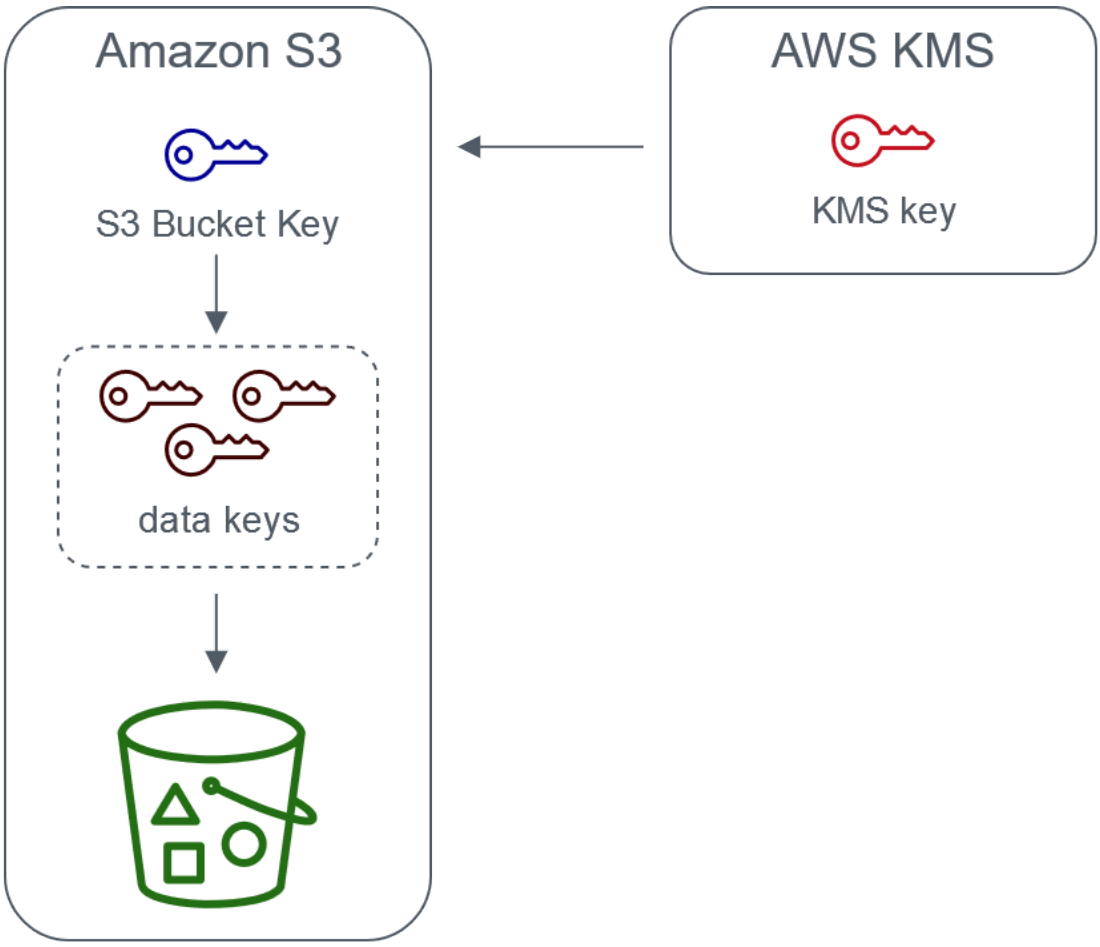
Wenn Sie Ihren Bucket für die Verwendung eines S3-Bucket-Schlüssels für SSE-KMS konfigurieren, AWS generiert einen kurzlebigen Schlüssel auf Bucket-Ebene aus AWS KMS und speichert ihn dann vorübergehend in S3. Dieser Schlüssel auf Bucket-Ebene erstellt während seines Lebenszyklus Datenschlüssel für neue Objekte. S3-Bucket-Schlüssel werden für einen begrenzten Zeitraum in Amazon S3 verwendet, wodurch S3 keine Anforderungen an stellen muss AWS KMS , um Verschlüsselungsvorgänge abzuschließen. Dadurch wird der Datenverkehr von S3 zu reduziert AWS KMS, sodass Sie zu einem Bruchteil der vorherigen Kosten auf AWS KMS-verschlüsselte Objekte in Amazon S3 zugreifen können.

Eindeutige Schlüssel auf Bucket-Ebene werden mindestens einmal pro Anforderer abgerufen, um sicherzustellen, dass der Zugriff des Anforderers auf den Schlüssel in einem AWS KMS CloudTrail Ereignis erfasst wird. Amazon S3 behandelt Aufrufer als unterschiedliche Anforderer, wenn sie unterschiedliche Rollen oder Konten oder dieselbe Rolle mit unterschiedlichen Umfangsrichtlinien verwenden. AWS KMS -Anforderungseinsparungen geben die Anzahl der Anforderer, Anforderungsmuster und das relative Alter der angeforderten Objekte an. Beispielsweise führt eine geringere Anzahl von Anforderern, die mehrere mit demselben Schlüssel auf Bucket-Ebene verschlüsselte Objekte in einem begrenzten Zeitfenster anfordern, zu größeren Einsparungen.

> **Note**
>
> Mit S3-Bucket-Schlüsseln können Sie AWS KMS Anforderungskosten sparen`Encrypt`, indem Sie Ihre Anforderungen an AWS KMS für `GenerateDataKey`-, - und -`Decrypt`Operationen durch die Verwendung eines Schlüssels auf Bucket-Ebene verringern. Nachfolgende Anforderungen, die diesen Schlüssel auf Bucket-Ebene nutzen, führen standardmäßig nicht zu AWS KMS API-Anforderungen und validieren den Zugriff nicht anhand der AWS KMS Schlüsselrichtlinie.

Wenn Sie einen S3-Bucket-Schlüssel konfigurieren, verwenden Objekte, die sich bereits im Bucket befinden, nicht den S3-Bucket-Schlüssel. Zum Konfigurieren eines S3-Bucket-Schlüssels für vorhandene Objekte können Sie eine `CopyObject`-Operation verwenden. Weitere Informationen finden Sie unter Konfigurieren eines S3-Bucket-Schlüssels auf Objektebene .

Amazon S3 gibt einen S3-Bucket-Schlüssel nur für Objekte frei, die mit demselben AWS KMS key verschlüsselt werden. S3-Bucket-Schlüssel sind mit KMS-Schlüsseln kompatibel, die von AWS KMS, importiertem Schlüsselmaterial und Schlüsselmaterial erstellt wurden, das von benutzerdefinierten Schlüsselspeichern unterstützt wird.



Server-side encryption with AWS Key Management service using an S3 Bucket Key

Konfigurieren von S3-Bucket-Schlüsseln

Sie können Ihren Bucket so konfigurieren, dass er einen S3-Bucket-Schlüssel für SSE-KMS bei neuen Objekten über die Amazon S3-Konsole AWS CLI, AWS SDKs oder REST API verwendet. Wenn S3-Bucket-Schlüssel in Ihrem Bucket aktiviert sind, verwenden Objekte, die mit einem anderen angegebenen SSE-KMS-Schlüssel hochgeladen wurden, ihre eigenen S3-Bucket-Schlüssel. Unabhängig von Ihrer S3-Bucket-Schlüsseleinstellung können Sie den Header `x-amz-server-side-encryption-bucket-key-enabled` mit einem `true`- oder `false`-Wert in Ihre Anforderung aufnehmen, um die Bucket-Einstellung zu überschreiben.

Bevor Sie Ihren Bucket für die Verwendung eines S3-Bucket-Schlüssels konfigurieren, lesen Sie Änderungen, die Sie vor dem Aktivieren eines S3-Bucket-Schlüssels beachten sollten.

Konfigurieren eines S3-Bucket-Schlüssels mit der Amazon-S3-Konsole

Wenn Sie einen neuen Bucket erstellen, können Sie ihn so konfigurieren, dass er einen S3-Bucket-Schlüssel für SSE-KMS bei neuen Objekten verwendet. Sie können einen vorhandenen Bucket auch dafür konfigurieren, dass er einen S3-Bucket-Schlüssel für SSE-KMS bei neuen Objekten verwendet, indem Sie die Bucket-Eigenschaften aktualisieren.

Weitere Informationen finden Sie unter Konfigurieren des Buckets für die Verwendung eines S3-Bucket-Schlüssels mit SSE-KMS bei neuen Objekten.

REST-API AWS CLI und AWS SDK-Unterstützung für S3-Bucket-Schlüssel

Sie können die REST-API oder das AWS -SDK verwenden AWS CLI, um Ihren Bucket für die Verwendung eines S3-Bucket-Schlüssels für SSE-KMS bei neuen Objekten zu konfigurieren. Sie können einen S3-Bucket-Schlüssel auch auf Objektebene aktivieren.

Weitere Informationen finden Sie hier:

- Konfigurieren eines S3-Bucket-Schlüssels auf Objektebene
- Konfigurieren des Buckets für die Verwendung eines S3-Bucket-Schlüssels mit SSE-KMS bei neuen Objekten

Die folgenden API-Operationen unterstützen S3-Bucket-Schlüssel für SSE-KMS:

- PutBucketEncryption
  - `ServerSideEncryptionRule` akzeptiert den Parameter `BucketKeyEnabled` zum Aktivieren und Deaktivieren eines S3-Bucket-Schlüssels.
- GetBucketEncryption
  - `ServerSideEncryptionRule` gibt die Einstellungen für `BucketKeyEnabled` zurück.
- PutObject, CopyObjectCreateMultipartUpload, und POST Object
  - Der Anforderungs-Header `x-amz-server-side-encryption-bucket-key-enabled` aktiviert oder deaktiviert einen S3-Bucket-Schlüssel auf Objektebene.
- HeadObject, GetObject, UploadPartCopyUploadPart, und CompleteMultipartUpload
  - Der Antwort-Header `x-amz-server-side-encryption-bucket-key-enabled` zeigt an, ob ein S3-Bucket-Schlüssel für ein Objekt aktiviert oder deaktiviert ist.

Arbeiten mit AWS CloudFormation

In enthält die `AWS::S3::Bucket` Ressource eine AWS CloudFormation Verschlüsselungseigenschaft namens `BucketKeyEnabled`, mit der Sie einen S3-Bucket-Schlüssel aktivieren oder deaktivieren können.

Weitere Informationen finden Sie unter Verwenden von AWS CloudFormation.

Änderungen, die Sie vor dem Aktivieren eines S3-Bucket-Schlüssels beachten sollten

Bevor Sie einen S3-Bucket-Schlüssel aktivieren, beachten Sie bitte die folgenden damit verbundenen Änderungen:

IAM- oder - AWS KMS Schlüsselrichtlinien

Wenn Ihre vorhandenen AWS Identity and Access Management (IAM)-Richtlinien oder AWS KMS Schlüsselrichtlinien Ihren Objekt-Amazon-Ressourcennamen (ARN) als Verschlüsselungskontext verwenden, um den Zugriff auf Ihren KMS-Schlüssel zu verfeinern oder einzuschränken, funktionieren diese Richtlinien nicht mit einem S3-Bucket-Schlüssel. S3-Bucket-Schlüssel verwenden den Bucket-ARN als Verschlüsselungskontext. Bevor Sie einen S3-Bucket-Schlüssel aktivieren, aktualisieren Sie Ihre IAM-Richtlinien oder AWS KMS Schlüsselrichtlinien, um Ihren Bucket-ARN als Verschlüsselungskontext zu verwenden.

Weitere Informationen zum Verschlüsselungskontext und zu S3-Bucket-Schlüsseln finden Sie unter Verschlüsselungskontext.

CloudTrail -Ereignisse für AWS KMS

Nachdem Sie einen S3-Bucket-Schlüssel aktiviert haben, protokollieren Ihre AWS KMS CloudTrail Ereignisse Ihren Bucket-ARN anstelle Ihres Objekt-ARN. Darüber hinaus werden in Ihren Protokollen weniger KMS- CloudTrail Ereignisse für SSE-KMS-Objekte angezeigt. Da das Schlüsselmaterial in Amazon S3 zeitlich begrenzt ist, werden weniger Anforderungen an gestellt AWS KMS.

Verwenden eines S3-Bucket-Schlüssels mit Replikation

Sie können S3-Bucket-Schlüssel mit Replikation innerhalb derselben Region (SRR) und regionsübergreifender Replikation (CRR) verwenden.

Wenn Amazon S3 ein verschlüsseltes Objekt repliziert, werden im Allgemeinen die Verschlüsselungseinstellungen des Replikatobjekts im Ziel-Bucket beibehalten. Wenn das Quellobjekt jedoch nicht verschlüsselt ist und Ihr Ziel-Bucket eine Standard-Verschlüsselung oder

einen S3-Bucket-Schlüssel verwendet, verschlüsselt Amazon S3 das Objekt mit der Konfiguration des Ziel-Buckets.

Die folgenden Beispiele veranschaulichen, wie ein S3-Bucket-Schlüssel mit der Replikation funktioniert. Weitere Informationen finden Sie unter Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten.

Example Beispiel 1 – Quellobjekt verwendet S3-Bucket-Schlüssel; Ziel-Bucket verwendet Standardverschlüsselung

Wenn Ihr Quellobjekt einen S3-Bucket-Schlüssel verwendet, Ihr Ziel-Bucket jedoch eine Standard-Verschlüsselung mit SSE-KMS, behält das Replikatobjekt seine S3-Bucket-Schlüssel-Verschlüsselungseinstellungen im Ziel-Bucket bei. Der Ziel-Bucket verwendet weiterhin die Standard-Verschlüsselung mit SSE-KMS.

Example Beispiel 2 – Quellobjekt ist nicht verschlüsselt; Ziel-Bucket verwendet einen S3-Bucket-Schlüssel mit SSE-KMS

Wenn Ihr Quellobjekt nicht verschlüsselt ist und der Ziel-Bucket einen S3-Bucket-Schlüssel mit SSE-KMS verwendet, wird das Replikatobjekt mit einem S3-Bucket-Schlüssel mit SSE-KMS im Ziel-Bucket verschlüsselt. Daher unterscheidet sich das `ETag` des Quell-Objekts von dem `ETag` des Replikatobjekts. Sie müssen die Anwendungen, die das `ETag` verwenden, zur Anpassung an diesen Unterschied aktualisieren.

Arbeiten mit S3-Bucket-Schlüsseln

Weitere Informationen zum Aktivieren und Arbeiten mit S3-Bucket-Schlüsseln finden Sie in den folgenden Abschnitten:

- Konfigurieren des Buckets für die Verwendung eines S3-Bucket-Schlüssels mit SSE-KMS bei neuen Objekten
- Konfigurieren eines S3-Bucket-Schlüssels auf Objektebene
- Anzeigen der Einstellungen für einen S3-Bucket-Schlüssel

Konfigurieren des Buckets für die Verwendung eines S3-Bucket-Schlüssels mit SSE-KMS bei neuen Objekten

Wenn Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) konfigurieren, können Sie Ihren Bucket so konfigurieren, dass er einen S3-

Bucket-Schlüssel für SSE-KMS für neue Objekte verwendet. S3-Bucket-Schlüssel verringern den Anforderungsverkehr von Amazon S3 zu AWS KMS und senken die Kosten für SSE-KMS. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

Sie können Ihren Bucket so konfigurieren, dass er einen S3-Bucket-Schlüssel für SSE-KMS für neue Objekte verwendet, indem Sie die Amazon S3-Konsole, die REST-API, AWS SDKs , AWS Command Line Interface (AWS CLI) oder verwenden AWS CloudFormation. Wenn Sie einen S3-Bucket-Schlüssel für vorhandene Objekte aktivieren oder deaktivieren möchten, können Sie eine `CopyObject`-Operation verwenden. Weitere Informationen finden Sie unter Konfigurieren eines S3-Bucket-Schlüssels auf Objektebene und Verwenden von S3-Batch-Vorgänge zum Verschlüsseln von Objekten mit S3-Bucket-Schlüssel.

Wenn ein S3-Bucket-Schlüssel für den Quell- oder Ziel-Bucket aktiviert ist, ist der Verschlüsselungskontext der Bucket-Amazon-Ressourcenname (ARN), nicht der Objekt-ARN, z. B., `arn:aws:s3:::`*`bucket_ARN`*. Sie müssen Ihre IAM-Richtlinien aktualisieren, um den Bucket-ARN für den Verschlüsselungskontext zu verwenden. Weitere Informationen finden Sie unter S3 Bucket-Schlüssel und Replikation.

Die folgenden Beispiele veranschaulichen, wie ein S3-Bucket-Schlüssel mit der Replikation funktioniert. Weitere Informationen finden Sie unter Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten.

Voraussetzungen

Bevor Sie Ihren Bucket für die Verwendung eines S3-Bucket-Schlüssels konfigurieren, lesen Sie Änderungen, die Sie vor dem Aktivieren eines S3-Bucket-Schlüssels beachten sollten.

Verwenden der S3-Konsole

In der S3-Konsole können Sie einen S3-Bucket-Schlüssel für einen neuen oder vorhandenen Bucket aktivieren oder deaktivieren. Objekte in der S3-Konsole übernehmen ihre S3-Bucket-Schlüssel-Einstellung aus der Bucket-Konfiguration. Wenn Sie einen S3-Bucket-Schlüssel für Ihren Bucket aktivieren, verwenden neue Objekte, die Sie in den Bucket hochladen, einen S3-Bucket-Schlüssel für SSE-KMS.

Hochladen, Kopieren oder Ändern von Objekten in Buckets, für die ein S3-Bucket-Schlüssel aktiviert ist

Wenn Sie ein Objekt in einen Bucket hochladen oder kopieren, für den ein S3-Bucket-Schlüssel aktiviert ist, oder ein Objekt darin ändern, werden die Einstellung des S3-Bucket-Schlüssels für dieses Objekt möglicherweise aktualisiert, um sie an die Bucket-Konfiguration anzupassen.

Wenn für ein Objekt bereits ein S3-Bucket-Schlüssel aktiviert ist, ändern sich die S3-Bucket-Schlüssel-Einstellungen für dieses Objekt nicht, wenn Sie das Objekt kopieren oder ändern. Wenn Sie jedoch ein Objekt ändern oder kopieren, für das kein S3-Bucket-Schlüssel aktiviert ist, und der Ziel-Bucket eine S3-Bucket-Schlüssel-Konfiguration hat, übernimmt das Objekt die S3-Bucket-Schlüssel-Einstellungen des Ziel-Buckets. Wenn beispielsweise für den Ziel-Bucket ein S3-Bucket-Schlüssel aktiviert ist, für das Quellobjekt jedoch nicht, wird ein S3-Bucket-Schlüssel für das Objekt aktiviert.

So aktivieren Sie einen S3-Bucket-Schlüssel beim Erstellen eines neuen Buckets

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.
4. Geben Sie Ihren Bucket-Namen ein und wählen Sie Ihre AWS-Region aus.
5. Wählen Sie unter Standardverschlüsselung für Verschlüsselungsschlüsseltyp die Option AWS Key Management Service -Schlüssel (SSE-KMS) aus.
6. Führen Sie unter AWS KMS -Schlüssel eine der folgenden Aktionen aus, um Ihren KMS-Schlüssel auszuwählen:

   - Um aus einer Liste der verfügbaren KMS-Schlüssel auszuwählen, wählen Sie Aus Ihrem AWS KMS keys auswählen und wählen Sie dann Ihren KMS-Schlüssel aus der Liste der verfügbaren Schlüssel aus.

     Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen zu kundenverwalteten Schlüsseln finden Sie unter Kundenschlüssel und - AWS Schlüssel im AWS Key Management Service -Entwicklerhandbuch.
   - Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS key -ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
   - Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

7. Wählen Sie unter Bucket Key (Bucket-Schlüssel) die Option Enable (Aktivieren).
8. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Amazon S3 erstellt Ihren Bucket mit einem aktivierten S3-Bucket-Schlüssel. Neue Objekte, die Sie in den Bucket hochladen, verwenden einen S3-Bucket-Schlüssel.

   Um einen S3-Bucket-Schlüssel zu deaktivieren, führen Sie die vorherigen Schritte aus und wählen Deaktivieren.

So aktivieren Sie einen S3-Bucket-Schlüssel für einen vorhandenen Bucket

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Bucket, für den Sie einen S3-Bucket-Schlüssel aktivieren möchten.
4. Wählen Sie die Registerkarte Eigenschaften aus.
5. Wählen Sie unter Default encryption (Standard-Verschlüsselung) Edit (Bearbeiten) aus.
6. Wählen Sie unter Standardverschlüsselung für Verschlüsselungsschlüsseltyp die Option AWS Key Management Service -Schlüssel (SSE-KMS) aus.
7. Führen Sie unter AWS KMS -Schlüssel eine der folgenden Aktionen aus, um Ihren KMS-Schlüssel auszuwählen:

   - Um aus einer Liste der verfügbaren KMS-Schlüssel auszuwählen, wählen Sie Aus Ihrem AWS KMS keys auswählen und wählen Sie dann Ihren KMS-Schlüssel aus der Liste der verfügbaren Schlüssel aus.

     Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen zu kundenverwalteten Schlüsseln finden Sie unter Kundenschlüssel und - AWS Schlüssel im AWS Key Management Service -Entwicklerhandbuch.

   - Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS key -ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.

- Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

  Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

8. Wählen Sie unter Bucket-Schlüssel die Option Aktivieren.
9. Wählen Sie Save Changes (Änderungen speichern).

   Amazon S3 aktiviert einen S3-Bucket-Schlüssel für neue Objekte, die zu Ihrem Bucket hinzugefügt werden. Bestehende Objekte verwenden den S3-Bucket-Schlüssel nicht. Zum Konfigurieren eines S3-Bucket-Schlüssels für vorhandene Objekte können Sie eine `CopyObject`-Operation verwenden. Weitere Informationen finden Sie unter Konfigurieren eines S3-Bucket-Schlüssels auf Objektebene .

   Um einen S3-Bucket-Schlüssel zu deaktivieren, führen Sie die vorherigen Schritte aus und wählen Deaktivieren.

Verwenden der REST-API

Sie können verwendenPutBucketEncryption, um einen S3-Bucket-Schlüssel für Ihren Bucket zu aktivieren oder zu deaktivieren. Um einen S3-Bucket-Schlüssel mit zu konfigurieren`PutBucketEncryption`, verwenden Sie den -ServerSideEncryptionRuleDatentyp, der die Standardverschlüsselung mit SSE-KMS beinhaltet. Sie können optional auch einen vom Kunden verwalteten Schlüssel verwenden, indem Sie die KMS-Schlüssel-ID für den kundenverwalteten Schlüssel angeben.

Weitere Informationen und Beispielsyntax finden Sie unter PutBucketEncryption.

Verwenden des AWS SDK for Java

Im folgenden Beispiel wird die Bucket-Standardverschlüsselung mit SSE-KMS und einem S3-Bucket-Schlüssel unter Verwendung des AWS SDK for Java aktiviert.

Java

```
AmazonS3 s3client = AmazonS3ClientBuilder.standard()
    .withRegion(Regions.DEFAULT_REGION)
    .build();
```

```
ServerSideEncryptionByDefault serverSideEncryptionByDefault = new
 ServerSideEncryptionByDefault()
    .withSSEAlgorithm(SSEAlgorithm.KMS);
ServerSideEncryptionRule rule = new ServerSideEncryptionRule()
    .withApplyServerSideEncryptionByDefault(serverSideEncryptionByDefault)
    .withBucketKeyEnabled(true);
ServerSideEncryptionConfiguration serverSideEncryptionConfiguration =
    new ServerSideEncryptionConfiguration().withRules(Collections.singleton(rule));

SetBucketEncryptionRequest setBucketEncryptionRequest = new
 SetBucketEncryptionRequest()
    .withServerSideEncryptionConfiguration(serverSideEncryptionConfiguration)
    .withBucketName(bucketName);

s3client.setBucketEncryption(setBucketEncryptionRequest);
```

Verwenden der AWS CLI

Im folgenden Beispiel wird die Bucket-Standardverschlüsselung mit SSE-KMS und einem S3-Bucket-Schlüssel unter Verwendung des AWS CLI aktiviert. Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3api put-bucket-encryption --bucket DOC-EXAMPLE-BUCKET --server-side-encryption-
configuration '{
        "Rules": [
            {
                "ApplyServerSideEncryptionByDefault": {
                    "SSEAlgorithm": "aws:kms",
                    "KMSMasterKeyID": "KMS-Key-ARN"
                },
                "BucketKeyEnabled": true
            }
        ]
    }'
```

Verwenden von AWS CloudFormation

Weitere Informationen zum Konfigurieren eines S3-Bucket-Schlüssels mit finden Sie AWS CloudFormation unter AWS::S3::Bucket ServerSideEncryptionRule im AWS CloudFormation - Benutzerhandbuch.

Konfigurieren eines S3-Bucket-Schlüssels auf Objektebene

Wenn Sie einen PUT- oder COPY-Vorgang mit der REST-API, den AWS SDKs oder ausführen, können Sie einen S3-Bucket-Schlüssel auf Objektebene aktivieren oder deaktivieren, indem Sie den `x-amz-server-side-encryption-bucket-key-enabled` Anforderungs-Header mit einem - `true` oder -`false`Wert hinzufügen. AWS CLI S3 S3-Bucket-Schlüssel reduzieren die Kosten für serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS) (SSE-KMS), indem der Anforderungsverkehr von Amazon S3 zu verringert wird AWS KMS. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

Wenn Sie einen S3-Bucket-Schlüssel für ein Objekt mithilfe eines PUT- oder COPY-Vorgangs konfigurieren, aktualisiert Amazon S3 nur die Einstellungen für dieses Objekt. Die S3-Bucket-Schlüssel-Einstellungen für den Ziel-Bucket ändern sich nicht. Wenn Sie eine PUT- oder COPY-Anforderung für ein KMS-verschlüsseltes Objekt in einen Bucket mit aktivierten S3-Bucket-Schlüsseln senden, verwendet Ihr Vorgang auf Objektebene automatisch S3-Bucket-Schlüssel, sofern Sie die Schlüssel im Anforderungs-Header nicht deaktivieren. Wenn Sie keinen S3-Bucket-Schlüssel für Ihr Objekt angeben, wendet Amazon S3 die S3-Bucket-Schlüssel-Einstellungen für den Ziel-Bucket auf das Objekt an.

Voraussetzung:

Bevor Sie Ihr Objekt für die Verwendung eines S3-Bucket-Schlüssels konfigurieren, lesen Sie Änderungen, die Sie vor dem Aktivieren eines S3-Bucket-Schlüssels beachten sollten.

Themen

- Amazon S3 Batchvorgänge
- Verwenden der REST-API
- Verwenden des AWS SDK for Java (PutObject)
- Verwenden der AWS CLI (PutObject)

Amazon S3 Batchvorgänge

Um Ihre vorhandenen Amazon-S3-Objekte zu verschlüsseln, können Sie Amazon-S3-Batchvorgänge verwenden. Sie stellen S3-Batchvorgänge eine Liste von Objekten bereit, für die Vorgänge ausgeführt werden sollen, und Batch-Vorgänge ruft die jeweilige API auf, um die angegebene Operation auszuführen.

Mit dem S3-Batch–Vorgangs-Kopiervorgang können Sie vorhandene nicht verschlüsselte Objekte kopieren und sie in denselben Bucket zurückschreiben, wie verschlüsselte Objekte. Ein einzelner Batch-Vorgangsauftrag kann die angegebene Operation für Milliarden von Objekten ausführen. Weitere Informationen finden Sie unter Ausführung umfangreicher Batch-Vorgänge für Amazon S3-Objekte durch. und Encrypting Objects with Amazon S3 Batch Operations (Verschlüsseln von Objekten mit Amazon S3 Batch Operations).

Verwenden der REST-API

Wenn Sie SSE-KMS verwenden, können Sie einen S3-Bucket-Schlüssel für ein Objekt mithilfe der folgenden API-Operationen aktivieren:

- PutObject – Wenn Sie ein Objekt hochladen, können Sie den `x-amz-server-side-encryption-bucket-key-enabled`Anforderungs-Header angeben, um einen S3-Bucket-Schlüssel auf Objektebene zu aktivieren oder zu deaktivieren.
- CopyObject – Wenn Sie ein Objekt kopieren und SSE-KMS konfigurieren, können Sie den `x-amz-server-side-encryption-bucket-key-enabled`Anforderungs-Header angeben, um einen S3-Bucket-Schlüssel für Ihr Objekt zu aktivieren oder zu deaktivieren.
- Post Object – Wenn Sie eine `POST`-Operation verwenden, um ein Objekt hochzuladen, und SSE-KMS konfigurieren, können Sie das Formularfeld `x-amz-server-side-encryption-bucket-key-enabled` verwenden, um einen S3-Bucket-Schlüssel für das Objekt zu aktivieren oder zu deaktivieren.
- CreateMultipartUpload – Wenn Sie große Objekte mithilfe der `CreateMultipartUpload` API-Operation hochladen und SSE-KMS konfigurieren, können Sie den `-x-amz-server-side-encryption-bucket-key-enabled`Anforderungs-Header verwenden, um einen S3-Bucket-Schlüssel für Ihr Objekt zu aktivieren oder zu deaktivieren.

Um einen S3-Bucket-Schlüssel auf Objektebene zu aktivieren, schließen Sie den Anforderungs-Header `x-amz-server-side-encryption-bucket-key-enabled` ein. Weitere Informationen über SSE-KMS und die REST API finden Sie unter Verwenden der REST-API.

Verwenden des AWS SDK for Java (PutObject)

Sie können das folgende Beispiel verwenden, um einen S3-Bucket-Schlüssel auf Objektebene mit AWS SDK for Java zu konfigurieren.

Java

```
AmazonS3 s3client = AmazonS3ClientBuilder.standard()
    .withRegion(Regions.DEFAULT_REGION)
    .build();

String bucketName = "DOC-EXAMPLE-BUCKET1";
String keyName = "key name for object";
String contents = "file contents";

PutObjectRequest putObjectRequest = new PutObjectRequest(bucketName, keyName,
 contents)
    .withBucketKeyEnabled(true);

s3client.putObject(putObjectRequest);
```

Verwenden der AWS CLI (PutObject)

Sie können das folgende AWS CLI Beispiel verwenden, um einen S3-Bucket-Schlüssel auf Objektebene als Teil einer -`PutObject`Anforderung zu konfigurieren.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key object key name --server-side-
encryption aws:kms --bucket-key-enabled --body filepath
```

Anzeigen der Einstellungen für einen S3-Bucket-Schlüssel

Sie können die Einstellungen für einen S3-Bucket-Schlüssel auf Bucket- oder Objektebene anzeigen, indem Sie die Amazon S3-Konsole, die REST-API, AWS Command Line Interface (AWS CLI) oder AWS SDKs verwenden.

S3-Bucket-Schlüssel verringern den Anforderungsverkehr von Amazon S3 zu AWS KMS und senken die Kosten für serverseitige Verschlüsselung mit AWS Key Management Service (SSE-KMS). Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

Zum Anzeigen der Einstellungen des S3-Bucket-Schlüssels für einen Bucket oder ein Objekt, der/das die Einstellungen des S3-Bucket-Schlüssels von der Bucket-Konfiguration übernommen hat, benötigen Sie die Berechtigung zum Ausführen der Aktion `s3:GetEncryptionConfiguration`.

Weitere Informationen finden Sie unter GetBucketEncryption in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der S3-Konsole

In der S3-Konsole können Sie die S3-Bucket-Schlüssel-Einstellungen für Ihren Bucket oder Ihr Objekt anzeigen. S3-Bucket-Schlüssel-Einstellungen werden aus der Bucket-Konfiguration übernommen, es sei denn, für die Quellobjekte ist bereits ein S3-Bucket-Schlüssel konfiguriert.

Objekte und Ordner im selben Bucket können unterschiedliche S3-Bucket-Schlüssel-Einstellungen haben. Wenn Sie beispielsweise ein Objekt über die REST-API hochladen und einen S3-Bucket-Schlüssel für das Objekt aktivieren, behält das Objekt seine S3-Bucket-Schlüssel-Einstellung im Ziel-Bucket bei, selbst wenn S3-Bucket-Schlüssel im Ziel-Bucket deaktiviert ist. Ein weiteres Beispiel: Wenn Sie einen S3-Bucket-Schlüssel für einen vorhandenen Bucket aktivieren, verwenden Objekte, die sich bereits im Bucket befinden, keinen S3-Bucket-Schlüssel. Für neue Objekte ist jedoch ein S3-Bucket-Schlüssel aktiviert.

So zeigen Sie die Einstellungen des S3-Bucket-Schlüssels für Ihren Bucket an

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Bucket, für den Sie einen S3-Bucket-Schlüssel aktivieren möchten.
4. Wählen Sie Properties (Eigenschaften).
5. Im Bereich Standard-Verschlüsselung sehen Sie unter Bucket-Schlüssel die S3-Bucket-Schlüssel-Einstellung für Ihren Bucket.

   Wenn Sie die S3-Bucket-Schlüssel-Einstellung nicht sehen, sind Sie möglicherweise nicht berechtigt, die `s3:GetEncryptionConfiguration`-Aktion durchzuführen. Weitere Informationen finden Sie unter GetBucketEncryption in der API-Referenz zu Amazon Simple Storage Service.

So zeigen Sie die S3-Bucket-Schlüssel-Einstellung für Ihr Objekt an

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie in der Liste Buckets den Bucket, für den Sie einen S3-Bucket-Schlüssel aktivieren möchten.
3. Wählen Sie in der Liste Objekte den Objektnamen aus.
4. Wählen Sie im Tab Details unter Serverseitige Verschlüsselungseinstellungen die Option Bearbeiten.

   Unter Bucket-Schlüssel sehen Sie die Einstellungen des S3-Bucket-Schlüssels für Ihr Objekt. Sie können diese Einstellungen nicht bearbeiten.

Verwenden der AWS CLI

So zeigen Sie die S3-Bucket-Schlüssel-Einstellungen auf Bucket-Ebene an

Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

```
aws s3api get-bucket-encryption --bucket DOC-EXAMPLE-BUCKET1
```

Weitere Informationen finden Sie unter get-bucket-encryption in der Referenz zum AWS CLI -Befehl.

So zeigen Sie die Einstellungen für einen S3-Bucket-Schlüssel auf Objektebene an

Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

```
aws s3api head-object --bucket DOC-EXAMPLE-BUCKET1 --key my_images.tar.bz2
```

Weitere Informationen finden Sie unter head-object in der AWS CLI -Befehlsreferenz.

Verwenden der REST-API

So zeigen Sie die S3-Bucket-Schlüssel-Einstellungen auf Bucket-Ebene an

Verwenden Sie die Operation `GetBucketEncryption`, um Verschlüsselungsinformationen für einen Bucket anzuzeigen, einschließlich der Einstellungen für einen S3-Bucket-Schlüssel. Die Einstellungen des S3-Bucket-Schlüssels werden im Antworttext im Element `ServerSideEncryptionConfiguration` mit der Einstellung `BucketKeyEnabled` angezeigt. Weitere Informationen finden Sie unter GetBucketEncryption in der Amazon-S3-API-Referenz.

So zeigen Sie die Einstellungen für einen S3-Bucket-Schlüssel auf Objektebene an

Verwenden Sie den Vorgang `HeadObject`, um den S3-Bucket-Schlüssel-Status für ein Objekt anzuzeigen. `HeadObject` liefert den Antwort-Header `x-amz-server-side-encryption-bucket-key-enabled`, der zeigt, ob ein S3-Bucket-Schlüssel für das Objekt aktiviert oder deaktiviert ist. Weitere Informationen finden Sie unter HeadObject in der Amazon-S3-API-Referenz.

Die folgenden API-Vorgänge geben auch den Antwort-Header `x-amz-server-side-encryption-bucket-key-enabled` zurück, wenn ein S3-Bucket-Schlüssel für ein Objekt konfiguriert ist:

- PutObject
- PostObject
- CopyObject
- CreateMultipartUpload
- UploadPartCopy
- UploadPart
- CompleteMultipartUpload
- GetObject

Verwenden der serverseitigen Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS)

Bei Verwendung der serverseitigen Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS) werden zwei Verschlüsselungsebenen auf Objekte angewendet, wenn sie in Amazon S3 hochgeladen werden. DSSE-KMS hilft Ihnen dabei, Compliance-Standards, die eine Multi-Layer-Verschlüsselung Ihrer Daten erfordern, einfacher zu erfüllen und die volle Kontrolle über Ihre Verschlüsselungsschlüssel zu haben.

Wenn Sie DSSE-KMS mit einem Amazon S3-Bucket verwenden, müssen sich die AWS KMS Schlüssel in derselben Region wie der Bucket befinden. Wird DSSE-KMS für das Objekt angefordert, wird außerdem die S3-Prüfsumme, die Teil der Objektmetadaten des Objekts ist, in verschlüsselter Form gespeichert. Weitere Informationen zu Prüfsummen finden Sie unter Überprüfung der Objektintegrität.

Für die Nutzung von DSSE-KMS und fallen zusätzliche Gebühren an AWS KMS keys. Weitere Informationen zu den DSSE-KMS-Preisen finden Sie unter AWS KMS key -Konzepte im AWS Key Management Service -Entwicklerhandbuch und in den AWS KMS -Preisen.

> **Note**
>
> S3-Bucket-Schlüssel werden für DSSE-KMS nicht unterstützt.

Erzwingen einer serverseitigen Dual-Layer-Verschlüsselung mit AWS KMS keys (DSSE-KMS)

Wenn Sie die serverseitige Dual-Layer-Verschlüsselung aller Objekte in einem bestimmten Amazon-S3-Bucket anfordern möchten, können Sie eine Bucket-Richtlinie verwenden. Beispielsweise verweigert die folgende Bucket-Richtlinie jedem die Berechtigung zum Hochladen von Objekten (`s3:PutObject`), wenn die Anforderung nicht den Header `x-amz-server-side-encryption` enthält, der die serverseitige Verschlüsselung mit DSSE-KMS anfordert.

```
{
            "Version":"2012-10-17",
            "Id":"PutObjectPolicy",
            "Statement":[{
                  "Sid":"DenyUnEncryptedObjectUploads",
                  "Effect":"Deny",
                  "Principal":"*",
                  "Action":"s3:PutObject",
                  "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
                  "Condition":{
                     "StringNotEquals":{
                        "s3:x-amz-server-side-encryption":"aws:kms:dsse"
                     }
                  }
               }
            ]
         }
```

Themen

- Angeben serverseitiger Dual-Layer-Verschlüsselung mit AWS KMS -Schlüsseln (DSSE-KMS)

Angeben serverseitiger Dual-Layer-Verschlüsselung mit AWS KMS -Schlüsseln (DSSE-KMS)

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon

S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Für alle Amazon-S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert und alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden im Ruhezustand automatisch verschlüsselt. Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) ist die Standardverschlüsselungskonfiguration für jeden Bucket in Amazon S3. Um einen anderen Verschlüsselungstyp zu verwenden, können Sie entweder die Art der serverseitigen Verschlüsselung angeben, die in Ihren S3-PUT-Anfragen verwendet werden soll, oder Sie können die Standardverschlüsselungskonfiguration im Ziel-Bucket festlegen.

Wenn Sie in Ihren PUT Anforderungen einen anderen Verschlüsselungstyp angeben möchten, können Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS), die serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) oder die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden. Wenn Sie im Ziel-Bucket eine andere Standardverschlüsselungskonfiguration festlegen möchten, können Sie SSE-KMS oder DSSE-KMS verwenden.

Sie können die Verschlüsselung anwenden, wenn Sie entweder ein neues Objekt hochladen oder ein vorhandenes Objekt kopieren.

Sie können DSSE-KMS mit der Amazon-S3-Konsole, Amazon-S3-REST-API und AWS Command Line Interface (AWS CLI) angeben. Weitere Informationen finden Sie unter den folgenden Themen.

Note

Sie können Multi-Region AWS KMS keys in Amazon S3 verwenden. Amazon S3 behandelt jedoch derzeit Multi-Regions-Schlüssel wie Einzel-Regions-Schlüssel und verwendet nicht die Multi-Regions-Funktionen des Schlüssels. Weitere Informationen finden Sie unter Using multi-Region keys (Verwenden von Multi-Regions-Zugriffpunkt-Schlüsseln) im AWS Key Management Service -Entwicklerhandbuch.

> **Note**
>
> Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen. Weitere Informationen zu kontoübergreifenden Berechtigungen für KMS-Schlüssel finden Sie unter Erstellen von KMS-Schlüsseln, die von anderen Konten verwendet werden können im Entwicklerhandbuch zu AWS Key Management Service .

Verwenden der S3-Konsole

In diesem Abschnitt wird beschrieben, wie Sie den Verschlüsselungstyp eines Objekts festlegen oder ändern, um die serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS) mithilfe der Amazon S3-Konsole zu verwenden.

> **Note**
>
> Wenn Sie die Verschlüsselungsmethode eines Objekts ändern, wird ein neues Objekt erstellt, um das alte zu ersetzen. Wenn S3-Versioning aktiviert ist, wird eine neue Version des Objekts erstellt, und das vorhandene Objekt wird zu einer älteren Version. Die Rolle, die die Eigenschaft ändert, wird auch Besitzer des neuen Objekts (oder der neuen Objektversion).

So fügen Sie die Verschlüsselung für ein Objekt hinzu oder ändern Sie sie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält, das Sie verschlüsseln möchten.
4. Wählen Sie in der Liste Objekte das Kontrollkästchen neben dem Objekt aus, für das Sie eine Verschlüsselung hinzufügen oder ändern möchten.

   Die Detailseite des Objekts wird angezeigt. Sie enthält mehrere Abschnitte mit den Eigenschaften des Objekts.
5. Wählen Sie die Registerkarte Eigenschaften aus.
6. Scrollen Sie nach unten zum Abschnitt Standardverschlüsselung und wählen Sie Bearbeiten aus.

Die Seite Standardverschlüsselung bearbeiten wird geöffnet.

7. Wählen Sie unter Verschlüsselungstyp die Option Serverseitige Dual-Layer-Verschlüsselung mit - AWS Key Management Service Schlüsseln (DSSE-KMS) aus.
8. Führen Sie unter AWS KMS -Schlüssel eine der folgenden Aktionen aus, um Ihren KMS-Schlüssel auszuwählen:

   - Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS keys wählen und dann den KMS-Schlüssel in der Liste der verfügbaren Schlüssel aus.

     Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS -Schlüssel im Entwicklerhandbuch zu AWS Key Management Service .
   - Um den KMS-Schlüssel-ARN einzugeben, wählen Sie AWS KMS key ARN eingeben und geben Sie dann Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
   - Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

     Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

> **Important**
>
> Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket verfügbar sind. Die Amazon-S3-Konsole führt nur die ersten 100 KMS-Schlüssel auf, die in derselben Region wie der Bucket verfügbar sind. Wenn Sie einen KMS-Schlüssel verwenden möchten, der nicht aufgeführt ist, müssen Sie den KMS-Schlüssel-ARN eingeben. Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen und Sie müssen den KMS-Schlüssel-ARN eingeben.
> Amazon S3 unterstützt nur symmetrisch verschlüsselte KMS-Schlüssel und keine asymmetrischen KMS-Schlüssel. Weitere Informationen finden Sie unter Erkennen von asymmetrischen KMS-Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service .

9. Wählen Sie unter Bucket-Schlüssel die Option Deaktivieren aus. S3-Bucket-Schlüssel werden für DSSE-KMS nicht unterstützt.
10. Wählen Sie Save Changes (Änderungen speichern).

> **Note**
>
> Diese Aktion wendet auf alle angegebenen Objekte Verschlüsselung an. Warten Sie beim Verschlüsseln von Ordnern, bis die Speicheroperation abgeschlossen ist, bevor Sie dem Ordner neue Objekte hinzufügen.

Verwenden der REST-API

Wenn Sie ein Objekt erstellen, d. h. wenn Sie ein neues Objekt hochladen oder ein vorhandenes Objekt kopieren, können Sie die Verwendung der serverseitigen Dual-Layer-Verschlüsselung mit AWS KMS keys (DSSE-KMS) angeben, um Ihre Daten zu verschlüsseln. Fügen Sie hierzu der Anforderung den Header `x-amz-server-side-encryption` hinzu. Setzen Sie den Wert des Headers auf den `aws:kms:dsse`-Verschlüsselungsalgorithmus. Amazon S3 bestätigt, dass Ihr Objekt unter Verwendung von DSSE-KMS-Verschlüsselung gespeichert wird, indem der Antwort-Header `x-amz-server-side-encryption` zurückgegeben wird.

Wenn Sie den Header `x-amz-server-side-encryption` mit dem Wert `aws:kms:dsse` angeben, können Sie auch die folgenden Anforderungs-Header verwenden:

- `x-amz-server-side-encryption: AES256 | aws:kms | aws:kms:dsse`
- `x-amz-server-side-encryption-aws-kms-key-id: SSEKMSKeyId`

Themen

- Amazon-S3-REST-API-Operationen, die DSSE-KMS unterstützen
- Verschlüsselungskontext (x-amz-server-side-encryption-context)
- AWS KMS -Schlüssel-ID (x-amz-server-side-encryption-aws-kms-key-id)

Amazon-S3-REST-API-Operationen, die DSSE-KMS unterstützen

Die folgenden REST-API-Vorgänge akzeptieren die Anforderungs-Header `x-amz-server-side-encryption`, `x-amz-server-side-encryption-aws-kms-key-id` und `x-amz-server-side-encryption-context`.

- PutObject – Wenn Sie Daten über die PUT-API-Operation hochladen, können Sie diese Anforderungs-Header angeben.
- CopyObject – Wenn Sie ein Objekt kopieren, erhalten Sie ein Quell- und ein Zielobjekt. Wenn Sie DSSE-KMS-Header mit der `CopyObject`-Operation übergeben, werden sie nur auf das Zielobjekt angewendet. Beim Kopieren eines vorhandenen Objekts wird das Zielobjekt unabhängig davon, ob das Quellobjekt verschlüsselt ist, nur dann verschlüsselt, wenn Sie die serverseitige Verschlüsselung explizit anfordern.
- POST-Objekt – Wenn Sie eine POST-Operation für das Hochladen eines Objekts verwenden, geben Sie die Informationen in die Formularfelder und nicht in die Anforderungs-Header ein.
- CreateMultipartUpload – Wenn Sie große Objekte über einen mehrteiligen Upload hochladen, können Sie diese Header in der `CreateMultipartUpload`-Anforderung angeben.

Die Antwort-Header der folgenden REST-API-Operationen geben den Header `x-amz-server-side-encryption` zurück, wenn ein Objekt mit der serverseitigen Verschlüsselung gespeichert wird.

- PutObject
- CopyObject
- POST-Objekt
- CreateMultipartUpload
- UploadPart
- UploadPartCopy
- CompleteMultipartUpload
- GetObject
- HeadObject

> **Important**
>
> - Alle - GET und -PUTAnfragen für ein Objekt, das durch geschützt ist, AWS KMS schlagen fehl, wenn Sie sie nicht mit Secure Sockets Layer (SSL), Transport Layer Security (TLS) oder Signature Version 4 erstellen.
> - Wenn Ihr Objekt DSSE-KMS verwendet, dürfen Sie keine Verschlüsselungsanforderungs-Header für GET- und HEAD-Anforderungen senden. Andernfalls erhalten Sie den Fehler HTTP 400 (Bad Request).

Verschlüsselungskontext (**`x-amz-server-side-encryption-context`**)

Wenn Sie `x-amz-server-side-encryption:aws:kms:dsse` angeben, unterstützt die Amazon-S3-API einen Verschlüsselungskontext mit dem Header `x-amz-server-side-encryption-context`. Ein Verschlüsselungskontext ist ein Satz von Schlüssel-Wert-Paaren, die zusätzliche kontextbezogene Informationen zu den Daten enthalten können.

Amazon S3 verwendet automatisch den Amazon-Ressourcennamen (ARN) des Objekts als Verschlüsselungskontextpaar, z. B. `arn:aws:s3:::`*`object_ARN`*.

Sie können optional ein zusätzliches Verschlüsselungskontextpaar bereitstellen, indem Sie den Header `x-amz-server-side-encryption-context` verwenden. Da der Verschlüsselungskontext jedoch nicht verschlüsselt ist, sollte er keine sensiblen Informationen enthalten. Amazon S3 speichert dieses zusätzliche Schlüsselpaar zusammen mit dem Standardverschlüsselungskontext.

Weitere Informationen zum Verschlüsselungskontext in Amazon S3 finden Sie unter Verschlüsselungskontext. Allgemeine Informationen zum Verschlüsselungs-Kontext finden Sie unter AWS Key Management Service Concepts – Encryption Context (Konzepte – Verschlüsselungs-Kontext) im AWS Key Management Service -Entwicklerhandbuch.

AWS KMS -Schlüssel-ID (**`x-amz-server-side-encryption-aws-kms-key-id`**)

Sie können den Header `x-amz-server-side-encryption-aws-kms-key-id` verwenden, um die ID des vom Kunden verwalteten Schlüssels anzugeben, der zum Schutz der Daten verwendet wird. Wenn Sie den `x-amz-server-side-encryption:aws:kms:dsse` Header angeben, aber nicht den `x-amz-server-side-encryption-aws-kms-key-id` Header, verwendet Amazon S3 die Von AWS verwalteter Schlüssel (`aws/s3`), um die Daten zu schützen. Wenn Sie einen vom

Kunden verwalteten Schlüssel verwenden möchten, müssen Sie den `x-amz-server-side-encryption-aws-kms-key-id`-Header des vom Kunden verwalteten Schlüssels angeben.

> **Important**
>
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung. Weitere Informationen zu diesen Schlüsseln finden Sie unter Symmetrische KMS-Verschlüsselungsschlüssel im Entwicklerhandbuch für AWS Key Management Service .

Verwenden der AWS CLI

Wenn Sie ein neues Objekt hochladen oder ein vorhandenes Objekt kopieren, können Sie für die Verschlüsselung Ihrer Daten DSSE-KMS angeben. Fügen Sie hierzu der Anforderung den Parameter `--server-side-encryption aws:kms:dsse` hinzu. Verwenden Sie den Parameter `--ssekms-key-id` *`example-key-id`*, um Ihren kundenverwalteten AWS KMS -Schlüssel hinzuzufügen, den Sie erstellt haben. Wenn Sie angeben`--server-side-encryption aws:kms:dsse`, aber keine -Schlüssel-ID angeben AWS KMS , verwendet Amazon S3 den - AWS verwalteten Schlüssel (`aws/s3`).

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key example-object-key --server-side-encryption aws:kms:dsse --ssekms-key-id example-key-id --body filepath
```

Sie können ein unverschlüsseltes Objekt mit DSSE-KMS verschlüsseln, indem Sie das Objekt wieder an seinen Platz kopieren.

```
aws s3api copy-object --bucket DOC-EXAMPLE-BUCKET --key example-object-key --
body filepath --bucket DOC-EXAMPLE-BUCKET --key example-object-key --sse aws:kms:dsse
 --sse-kms-key-id example-key-id --body filepath
```

Verwenden von serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Die serverseitige Verschlüsselung dient zum Schutz ruhender Daten. Die serverseitige Verschlüsselung verschlüsselt nur die Objektdaten, nicht die Metadaten des Objekts. Die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) gestattet Ihnen, eigene Verschlüsselungsschlüssel zu speichern. Mit dem Verschlüsselungsschlüssel, den Sie als

Teil Ihrer Anforderung bereitstellen, verwaltet Amazon S3 die Datenverschlüsselung, wenn es auf Datenträger schreibt, und die Entschlüsselung, wenn Sie auf Ihre Objekte zugreifen. Sie müssen also für die Datenverschlüsselung und -entschlüsselung keinen Code mehr verwalten. Sie müssen nur noch die von Ihnen bereitgestellten Verschlüsselungsschlüssel verwalten.

Wenn Sie ein Objekt hochladen, verwendet Amazon S3 den von Ihnen bereitgestellten Verschlüsselungsschlüssel, um eine AES-256-Verschlüsselung auf Ihre Daten anzuwenden. Amazon S3 entfernt dann den Verschlüsselungsschlüssel aus dem Speicher. Wenn Sie ein Objekt abrufen, müssen Sie denselben Verschlüsselungsschlüssel als Teil Ihrer Anfrage angeben. Amazon S3 überprüft zuerst, ob der von Ihnen bereitgestellte Verschlüsselungsschlüssel übereinstimmt, und entschlüsselt das Objekt, bevor Objektdaten zurückgegeben werden.

Für die Nutzung von SSE-C fallen keine zusätzlichen Gebühren an. Für Anforderungen zum Konfigurieren und Verwenden von SSE-C werden jedoch Standardgebühren für Amazon-S3-Anforderungen berechnet. Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

> **Note**
>
> Amazon S3 speichert den von Ihnen bereitgestellten Verschlüsselungsschlüssel nicht. Stattdessen wird ein zufällig mit einem Salt versehener Hash-basierter Nachrichtenauthentifizierungscode (HMAC) des Verschlüsselungsschlüssels gespeichert, um zukünftige Anfragen zu überprüfen. Der mit einem Salt versehene HMAC-Wert kann nicht verwendet werden, um den Wert des Verschlüsselungsschlüssels abzuleiten oder den Inhalt des verschlüsselten Objekts zu entschlüsseln. Das bedeutet, wenn Sie den Verschlüsselungsschlüssel verlieren, verlieren Sie das Objekt.

Die S3-Replikation unterstützt Objekte, die mit SSE-C verschlüsselt sind. Weitere Informationen zum Replizieren verschlüsselter Objekte finden Sie unter the section called “Replizieren von verschlüsselten Objekten (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS)”.

Weitere Informationen zu SSE-C finden Sie in den folgenden Themen.

Themen

- Übersicht über SSE-C
- Anfordern und Einschränken von SSE-C
- Vorsignierte URLs und SSE-C
- Angeben der serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Übersicht über SSE-C

Dieser Abschnitt bietet eine Übersicht über SSE-C. Bei der Verwendung von SSE-C sind die folgenden Punkte zu beachten.

- Sie müssen HTTPS verwenden.

  > **Important**
  >
  > Amazon S3 weist Anfragen über HTTP zurück, wenn SSE-C verwendet wird. Aus Sicherheitsgründen sollten Sie jeden Schlüssel, den Sie versehentlich mittels HTTP senden, als nicht vertrauenswürdig betrachten. Verwerfen Sie den Schlüssel und rotieren Sie ihn wie erforderlich.

- Das Entity-Tag (ETag) in der Antwort ist nicht der MD5-Hash der Objektdaten.
- Sie Verwalten ein Mapping, welcher Verschlüsselungsschlüssel für die Verschlüsselung welches Objekts verwendet wurde. Amazon S3 speichert keine Verschlüsselungsschlüssel. Sie sind dafür verantwortlich, zu verwalten, welchen Verschlüsselungsschlüssel Sie für welches Objekt angegeben haben.
  - Wenn für Ihren Bucket Versioning aktiviert ist, kann jede Objektversion, die Sie mit dieser Funktion hochladen, einen eigenen Verschlüsselungsschlüssel haben. Sie sind dafür verantwortlich, zu verwalten, welcher Verschlüsselungsschlüssel für welche Objektversion verwendet wurde.
  - Sie verwalten die Verschlüsselungsschlüssel auf der Clientseite, deshalb verwalten Sie auch alle zusätzlichen Sicherungsmechanismen auf der Clientseite, wie beispielsweise die Schlüsselrotation.

  > **Warning**
  >
  > Wenn Sie den Verschlüsselungsschlüssel verlieren, schlagen alle `GET`-Anfragen für ein Objekt ohne seinen Verschlüsselungsschlüssel fehl und Sie verlieren das Objekt.

Anfordern und Einschränken von SSE-C

Um SSE-C für alle Objekte in einem bestimmten Amazon-S3-Bucket anzufordern, können Sie eine Bucket-Richtlinie verwenden.

Die folgende Bucket-Richtlinie verweigert beispielsweise die Berechtigung zum Hochladen von Objekten (`s3:PutObject`) für alle Anfragen, die nicht den Header `x-amz-server-side-encryption-customer-algorithm` enthalten, der SSE-C anfordert.

```
{
    "Version": "2012-10-17",
    "Id": "PutObjectPolicy",
    "Statement": [
        {
            "Sid": "RequireSSECObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption-customer-algorithm": "true"
                }
            }
        }
    ]
}
```

Sie können auch eine Richtlinie verwenden, um die serverseitige Verschlüsselung aller Objekte in einem bestimmten Amazon-S3-Bucket einzuschränken. Beispielsweise verweigert die folgende Bucket-Richtlinie jedem die `s3:PutObject`-Berechtigung zum Hochladen von Objekten, wenn die Anfrage nicht den `x-amz-server-side-encryption-customer-algorithm`-Header enthält, der SSE-C anfordert.

```
{
    "Version": "2012-10-17",
    "Id": "PutObjectPolicy",
    "Statement": [
        {
            "Sid": "RestrictSSECObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "Null": {
```

```
                    "s3:x-amz-server-side-encryption-customer-algorithm": "false"
                }
            }
        }
    ]
}
```

> **Important**
>
> Wenn Sie eine Bucket-Richtlinie verwenden, um SSE-C für zu erzwingen`s3:PutObject`, müssen Sie den `-x-amz-server-side-encryption-customer-algorithm`Header in alle mehrteiligen Upload-Anforderungen (CreateMultipartUpload UploadPart, und CompleteMultipartUpload) aufnehmen.

Vorsignierte URLs und SSE-C

Sie können eine vorsignierte URL erstellen, die für Operationen wie das Hochladen eines neuen Objekts, das Abrufen eines vorhandenen Objekts oder von Objekt-Metadaten verwendet werden kann. Vorsignierte URLs unterstützen die SSE-C wie folgt:

- Beim Erstellen einer vorsignierten URL müssen Sie den Algorithmus unter Verwendung des `x-amz-server-side-encryption-customer-algorithm` -Headers in der Signaturberechnung angeben.
- Wenn Sie die vorsignierte URL verwenden, um ein neues Objekt hochzuladen, ein vorhandenes Objekt abzurufen oder nur Objekt-Metadaten abzurufen, müssen Sie in der Anforderung Ihrer Client-Anwendung alle Verschlüsselungs-Header angeben.

> **Note**
>
> Für andere Objekte als SSE-C-Objekte können Sie eine vorsignierte URL generieren und diese URL direkt in einen Browser kopieren, um beispielsweise auf die Daten zuzugreifen. Dies ist für SSE-C-Objekte jedoch nicht möglich, da Sie zusätzlich zur vorsignierten URL auch HTTP-Header einfügen müssen, die für SSE-C-Objekte spezifisch sind. Daher können Sie vorsignierte URLs für SSE-C-Objekte ausschließlich programmgesteuert verwenden.

Weitere Informationen zu vorsignierten URLs finden Sie unter the section called “Arbeiten mit vorsignierten URLs”.

Angeben der serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Zum Zeitpunkt der Objekterstellung über die REST-API können Sie die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) angeben. Wenn Sie SSE-C verwenden, müssen Sie mithilfe der folgenden Anforderungs-Header Informationen zum Verschlüsselungsschlüssel bereitstellen.

Name	Beschreibung
`x-amz-server-side-encryption-customer-algorithm`	Verwenden Sie diesen Header, um den Verschlüsselungsalgorithmus anzugeben. Der Header-Wert muss `AES256` sein.
`x-amz-server-side-encryption-customer-key`	Verwenden Sie diesen Header, um den base64-codierten 256-bit-V erschlüsselungsschlüssel für Amazon S3 bereitzustellen, mit dem Ihre Daten verschlüsselt und entschlüsselt werden.
`x-amz-server-side-encryption-customer-key-MD5`	Verwenden Sie diesen Header, um den base64-codierten 128-bit-Verschlüsselungsschlüssel für Amazon S3 bereitzustellen, mit dem Ihre Daten gemäß RFC 1321 verschlüsselt und entschlüsselt werden. Amazon S3 verwendet diesen Header für eine Überprüfung der Nachrichtenintegrität, um sicherzustellen, dass der Verschlüsselungsschlüssel fehlerfrei übertragen wurde.

Sie können AWS SDK-Wrapper-Bibliotheken verwenden, um diese Header zu Ihrer Anfrage hinzuzufügen. Falls nötig, können Sie auch die REST-API-Aufrufe in Amazon S3 direkt von Ihrer Anwendung aus durchführen.

> **Note**
>
> Sie können die Amazon-S3-Konsole nicht zum Hochladen eines Objekts und zum Anfordern von SSE-C verwenden. Sie können die Konsole auch nicht verwenden, um ein vorhandenes

Objekt zu aktualisieren (beispielsweise durch Ändern der Speicherklasse oder Hinzufügen von Metadaten), das mittels SSE-C gespeichert wurde.

Verwenden der REST-API

Amazon-S3-Rest-APIs, die SSE-C unterstützen

Die folgenden Amazon-S3-APIs unterstützen die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C).

- GET-Operation – Sie können beim Abrufen von Objekten über die GET-API (siehe GET Object) die Anforderungs-Header angeben.
- HEAD-Operation – Um Objektmetadaten über die HEAD-API abzurufen (siehe HEAD Object), können Sie diese Anforderungs-Header angeben.
- Operation PUT: Sie können beim Hochladen von Daten über die PUT Object API (siehe PUT Object) diese Anforderungs-Header angeben.
- Multipart Upload – Sie können beim Hochladen großer Objekte über die Multipart Upload-API diese Header angeben. Sie geben diese Header in der Initiierungsanforderung (siehe Initiieren mehrteiliger Uploads) und in jeder nachfolgenden Anforderung für einen Teil-Upload an (siehe Hochladen von Teilen oder Hochladen von Teilen – Kopieren). Für jede teilweise Upload-Anfrage muss dieselbe Verschlüsslungsinformation angegeben werden, wie diejenige, die Sie in der Initiierungsanfrage für den mehrteiligen Upload angegeben haben.
- POST-Operation – Bei Verwendung einer POST-Operation zum Hochladen eines Objekts (siehe POST Object) geben Sie die Informationen in den Formularfeldern und nicht in den Anforderungs-Headern an.
- Copy-Operation – Wenn Sie ein Objekt kopieren (siehe PUT Object – Copy), erhalten Sie ein Quell- und ein Zielobjekt:
  - Wenn Sie möchten, dass das Zielobjekt mit serverseitiger Verschlüsselung mit AWS verwalteten Schlüsseln verschlüsselt wird, müssen Sie den `x-amz-server-side-encryption` Anforderungs-Header angeben.
  - Wenn das Zielobjekt unter Verwendung von SSE-C verschlüsselt werden soll, müssen Sie die Verschlüsselungs-Informationen unter Verwendung der drei Header angeben, die in der obigen Tabelle beschrieben sind.

- Wenn das Quellobjekt unter Verwendung von SSE-C verschlüsselt ist, müssen Sie die Verschlüsselungsschlüsselinformationen unter Verwendung der folgenden Header angeben, sodass Amazon S3 das Objekt zum Kopieren entschlüsseln kann.

Name	Beschreibung
`x-amz-copy-source-server-side-encryption-customer-algorithm`	Verwenden Sie diesen Header, um den Verschlüsselungsalgorithmus anzugeben, den Amazon S3 für die Entschlüsselung des Quellobjekts verwenden soll. Dieser Wert muss `AES256` lauten.
`x-amz-copy-source-server-side-encryption-customer-key`	Verwenden Sie diesen Header, um den base64-codierten 256-bit-Verschlüsselungsschlüssel für Amazon S3 bereitzustellen, mit dem das Quellobjekt entschlüsselt werden soll. Dieser Verschlüsselungsschlüssel muss derjenige sein, den Sie Amazon S3 beim Erstellen des Quellobjekts bereitgestellt haben. Andernfalls kann Amazon S3 das Objekt nicht entschlüsseln.
`x-amz-copy-source-server-side-encryption-customer-key-MD5`	Verwenden Sie diesen Header, um den base64-codierten 128-bit-MD5-Digest des Verschlüsselungsschlüssels gemäß RFC 1321 bereitzustellen.

Verwenden der - AWS SDKs zur Angabe von SSE-C für PUT-, GET-, Head- und Copy-Operationen

In den folgenden Beispielen wird gezeigt, wie Sie für Objekte die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) anfordern. Dabei werden folgende Vorgänge ausgeführt. Jede Operation zeigt, wie SSE-C-bezogene Header in der Anfrage angegeben werden:

- Put object – Lädt ein Objekt hoch und fordert die serverseitige Verschlüsselung mit einem vom Kunden bereitgestellten Verschlüsselungsschlüssel an.
- Get object – Lädt das im vorigen Schritt hochgeladene Objekt herunter. Sie stellen in der Anforderung dieselben Verschlüsselungs-Informationen bereit, die Sie beim Hochladen des Objekts angegeben haben. Amazon S3 benötigt diese Informationen, um das Objekt zu entschlüsseln und an Sie zurückzugeben.

- Get object metadata – Ruft die Metadaten des Objekts ab. Sie stellen dieselben Verschlüsselungs-Informationen bereit, die beim Erstellen des Objekts verwendet wurden.
- Copy object – Erstellt eine Kopie des zuvor hochgeladenen Objekts. Da das Quellobjekt mit SSE-C gespeichert wurde, müssen Sie seine Verschlüsselungs-Informationen in Ihrer Kopieranfrage bereitstellen. Standardmäßig verschlüsselt Amazon S3 die Kopie des Objekts nur, wenn Sie dies ausdrücklich anfordern. In diesem Beispiel wird Amazon S3 dazu angewiesen, eine verschlüsselte Kopie des Objekts zu speichern.

Java

> **Note**
>
> Dieses Beispiel zeigt, wie ein Objekt in einer einzigen Operation hochgeladen wird. Wenn Sie die API für mehrteilige Uploads verwenden, um große Objekte hochzuladen, geben Sie Verschlüsslungsinformationen genauso wie in diesem Beispiel veranschaulicht ein. Beispiele für mehrteilige Uploads, die die verwenden AWS SDK for Java, finden Sie unter Hochladen eines Objekts mit Multipart-Upload.

Um die erforderlichen Verschlüsselungsinformationen hinzuzufügen, schließen Sie in Ihre Anfrage einen `SSECustomerKey` ein. Weitere Informationen zur Klasse `SSECustomerKey` finden Sie im Abschnitt "REST API".

Weitere Informationen über SSE-C finden Sie unter Verwenden von serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;
```

```
import javax.crypto.KeyGenerator;
import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStreamReader;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;

public class ServerSideEncryptionUsingClientSideEncryptionKey {
    private static SSECustomerKey SSE_KEY;
    private static AmazonS3 S3_CLIENT;
    private static KeyGenerator KEY_GENERATOR;

    public static void main(String[] args) throws IOException,
 NoSuchAlgorithmException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String uploadFileName = "*** File path ***";
        String targetKeyName = "*** Target key name ***";

        // Create an encryption key.
        KEY_GENERATOR = KeyGenerator.getInstance("AES");
        KEY_GENERATOR.init(256, new SecureRandom());
        SSE_KEY = new SSECustomerKey(KEY_GENERATOR.generateKey());

        try {
            S3_CLIENT = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Upload an object.
            uploadObject(bucketName, keyName, new File(uploadFileName));

            // Download the object.
            downloadObject(bucketName, keyName);

            // Verify that the object is properly encrypted by attempting to
 retrieve it
            // using the encryption key.
            retrieveObjectMetadata(bucketName, keyName);

            // Copy the object into a new object that also uses SSE-C.
```

```
            copyObject(bucketName, keyName, targetKeyName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void uploadObject(String bucketName, String keyName, File file) {
        PutObjectRequest putRequest = new PutObjectRequest(bucketName, keyName,
file).withSSECustomerKey(SSE_KEY);
        S3_CLIENT.putObject(putRequest);
        System.out.println("Object uploaded");
    }

    private static void downloadObject(String bucketName, String keyName) throws
IOException {
        GetObjectRequest getObjectRequest = new GetObjectRequest(bucketName,
keyName).withSSECustomerKey(SSE_KEY);
        S3Object object = S3_CLIENT.getObject(getObjectRequest);

        System.out.println("Object content: ");
        displayTextInputStream(object.getObjectContent());
    }

    private static void retrieveObjectMetadata(String bucketName, String keyName) {
        GetObjectMetadataRequest getMetadataRequest = new
GetObjectMetadataRequest(bucketName, keyName)
                .withSSECustomerKey(SSE_KEY);
        ObjectMetadata objectMetadata =
S3_CLIENT.getObjectMetadata(getMetadataRequest);
        System.out.println("Metadata retrieved. Object size: " +
objectMetadata.getContentLength());
    }

    private static void copyObject(String bucketName, String keyName, String
targetKeyName)
            throws NoSuchAlgorithmException {
        // Create a new encryption key for target so that the target is saved using
        // SSE-C.
```

```
        SSECustomerKey newSSEKey = new SSECustomerKey(KEY_GENERATOR.generateKey());

        CopyObjectRequest copyRequest = new CopyObjectRequest(bucketName, keyName,
 bucketName, targetKeyName)
                .withSourceSSECustomerKey(SSE_KEY)
                .withDestinationSSECustomerKey(newSSEKey);

        S3_CLIENT.copyObject(copyRequest);
        System.out.println("Object copied");
    }

    private static void displayTextInputStream(S3ObjectInputStream input) throws
 IOException {
        // Read one line at a time from the input stream and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
        String line;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

.NET

> **Note**
>
> Beispiele für das Hochladen großer Objekte mithilfe der API für mehrteilige Uploads finden Sie unter Hochladen eines Objekts mit Multipart-Upload und Verwenden der - AWS SDKs (Low-Level-API).

Weitere Informationen über SSE-C finden Sie unter Verwenden von serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C). Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

Example

```
using Amazon;
using Amazon.S3;
```

```
using Amazon.S3.Model;
using System;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class SSEClientEncryptionKeyObjectOperationsTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for new object created ***";
        private const string copyTargetKeyName = "*** key name for object copy ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            ObjectOpsUsingClientEncryptionKeyAsync().Wait();
        }
        private static async Task ObjectOpsUsingClientEncryptionKeyAsync()
        {
            try
            {
                // Create an encryption key.
                Aes aesEncryption = Aes.Create();
                aesEncryption.KeySize = 256;
                aesEncryption.GenerateKey();
                string base64Key = Convert.ToBase64String(aesEncryption.Key);

                // 1. Upload the object.
                PutObjectRequest putObjectRequest = await
 UploadObjectAsync(base64Key);
                // 2. Download the object and verify that its contents matches what
 you uploaded.
                await DownloadObjectAsync(base64Key, putObjectRequest);
                // 3. Get object metadata and verify that the object uses AES-256
 encryption.
                await GetObjectMetadataAsync(base64Key);
                // 4. Copy both the source and target objects using server-side
 encryption with
```

```
                //    a customer-provided encryption key.
                await CopyObjectAsync(aesEncryption, base64Key);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing
an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        private static async Task<PutObjectRequest> UploadObjectAsync(string
base64Key)
        {
            PutObjectRequest putObjectRequest = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
                ContentBody = "sample text",
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };
            PutObjectResponse putObjectResponse = await
client.PutObjectAsync(putObjectRequest);
            return putObjectRequest;
        }
        private static async Task DownloadObjectAsync(string base64Key,
PutObjectRequest putObjectRequest)
        {
            GetObjectRequest getObjectRequest = new GetObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
                // Provide encryption information for the object stored in Amazon
S3.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };
```

```
            using (GetObjectResponse getResponse = await
client.GetObjectAsync(getObjectRequest))
            using (StreamReader reader = new
StreamReader(getResponse.ResponseStream))
            {
                string content = reader.ReadToEnd();
                if (String.Compare(putObjectRequest.ContentBody, content) == 0)
                    Console.WriteLine("Object content is same as we uploaded");
                else
                    Console.WriteLine("Error...Object content is not same.");

                if (getResponse.ServerSideEncryptionCustomerMethod ==
ServerSideEncryptionCustomerMethod.AES256)
                    Console.WriteLine("Object encryption method is AES256, same as
we set");
                else
                    Console.WriteLine("Error...Object encryption method is not the
same as AES256 we set");

                // Assert.AreEqual(putObjectRequest.ContentBody, content);
                // Assert.AreEqual(ServerSideEncryptionCustomerMethod.AES256,
getResponse.ServerSideEncryptionCustomerMethod);
            }
        }
        private static async Task GetObjectMetadataAsync(string base64Key)
        {
            GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest
            {
                BucketName = bucketName,
                Key = keyName,

                // The object stored in Amazon S3 is encrypted, so provide the
necessary encryption information.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };

            GetObjectMetadataResponse getObjectMetadataResponse = await
client.GetObjectMetadataAsync(getObjectMetadataRequest);
            Console.WriteLine("The object metadata show encryption method used is:
{0}", getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
```

```
            // Assert.AreEqual(ServerSideEncryptionCustomerMethod.AES256,
 getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
        }
        private static async Task CopyObjectAsync(Aes aesEncryption, string
 base64Key)
        {
            aesEncryption.GenerateKey();
            string copyBase64Key = Convert.ToBase64String(aesEncryption.Key);

            CopyObjectRequest copyRequest = new CopyObjectRequest
            {
                SourceBucket = bucketName,
                SourceKey = keyName,
                DestinationBucket = bucketName,
                DestinationKey = copyTargetKeyName,
                // Information about the source object's encryption.
                CopySourceServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                CopySourceServerSideEncryptionCustomerProvidedKey = base64Key,
                // Information about the target object's encryption.
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = copyBase64Key
            };
            await client.CopyObjectAsync(copyRequest);
        }
    }
}
```

Verwenden der - AWS SDKs zur Angabe von SSE-C für mehrteilige Uploads

Das Beispiel im obigen Abschnitt zeigt, wie Sie eine serverseitige Verschlüsselung mit einem vom Kunden bereitgestellten Schlüssel (SSE-C) in den PUT-, GET-, Head- und Copy-Vorgänge anfordern. In diesem Abschnitt werden andere Amazon-S3-APIs beschrieben, die SSE-C unterstützen.

Java

Um große Objekte hochzuladen, können Sie die API für mehrteilige Uploads verwenden (siehe Hochladen und Kopieren von Objekten mit mehrteiligen Uploads). Sie können entweder High-Level- und Low-Level-APIs zum Hochladen großer Objekte verwenden. Diese APIs unterstützen verschlüsselungsbezogene Header in der Anforderung.

- Wenn Sie die High-Level-`TransferManager`-API verwenden, geben Sie die verschlüsselungsspezifischen Header in der `PutObjectRequest` an (siehe Hochladen eines Objekts mit Multipart-Upload).
- Beim Verwenden der Low-Level-API geben Sie verschlüsselungsbezogene Informationen in der `InitiateMultipartUploadRequest` gefolgt von identischen Verschlüsselungsinformationen in jeder `UploadPartRequest` an. In Ihrer `CompleteMultipartUploadRequest` müssen Sie keine verschlüsselungsspezifischen Header angeben. Beispiele finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

Der folgende Beispiel verwendet den `TransferManager`, um Objekte zu erstellen, und zeigt, wie Sie SSE-C-bezogene Informationen bereitstellen. Das Beispiel erledigt Folgendes:

- Erstellt ein Objekts mit der Methode `TransferManager.upload()`. In der Instance `PutObjectRequest` geben Sie anzufordernde Verschlüsselungsschlüssel-Informationen an. Amazon S3 verschlüsselt das Objekt mit dem vom Kunden bereitgestellten Schüssel.
- Erstellt eine Kopie des Objekts durch Aufrufen der `TransferManager.copy()`-Methode. Das Beispiel weist Amazon S3 dazu an, eine Objektkopie unter Verwendung eines neuen `SSECustomerKey` zu verschlüsseln. Da das Quellobjekt mittels SSE-C verschlüsselt ist, stellt die `CopyObjectRequest` auch den Verschlüsselungsschlüssel des Quellobjekts bereit. Auf diese Weise kann Amazon S3 das Objekt vor dem Kopieren entschlüsseln.

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;
import com.amazonaws.services.s3.model.PutObjectRequest;
import com.amazonaws.services.s3.model.SSECustomerKey;
import com.amazonaws.services.s3.transfer.Copy;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;
import com.amazonaws.services.s3.transfer.Upload;
```

```
import javax.crypto.KeyGenerator;
import java.io.File;
import java.security.SecureRandom;

public class ServerSideEncryptionCopyObjectUsingHLwithSSEC {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String fileToUpload = "*** File path ***";
        String keyName = "*** New object key name ***";
        String targetKeyName = "*** Key name for object copy ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();

            // Create an object from a file.
            PutObjectRequest putObjectRequest = new PutObjectRequest(bucketName,
 keyName, new File(fileToUpload));

            // Create an encryption key.
            KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
            keyGenerator.init(256, new SecureRandom());
            SSECustomerKey sseCustomerEncryptionKey = new
 SSECustomerKey(keyGenerator.generateKey());

            // Upload the object. TransferManager uploads asynchronously, so this
 call
            // returns immediately.
            putObjectRequest.setSSECustomerKey(sseCustomerEncryptionKey);
            Upload upload = tm.upload(putObjectRequest);

            // Optionally, wait for the upload to finish before continuing.
            upload.waitForCompletion();
            System.out.println("Object created.");

            // Copy the object and store the copy using SSE-C with a new key.
```

```
            CopyObjectRequest copyObjectRequest = new CopyObjectRequest(bucketName,
 keyName, bucketName, targetKeyName);
            SSECustomerKey sseTargetObjectEncryptionKey = new
 SSECustomerKey(keyGenerator.generateKey());
            copyObjectRequest.setSourceSSECustomerKey(sseCustomerEncryptionKey);

 copyObjectRequest.setDestinationSSECustomerKey(sseTargetObjectEncryptionKey);

            // Copy the object. TransferManager copies asynchronously, so this call
 returns
            // immediately.
            Copy copy = tm.copy(copyObjectRequest);

            // Optionally, wait for the upload to finish before continuing.
            copy.waitForCompletion();
            System.out.println("Copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Um große Objekte hochzuladen, können Sie die API für mehrteilige Uploads verwenden (siehe Hochladen und Kopieren von Objekten mit mehrteiligen Uploads). AWS SDK for .NET bietet sowohl High-Level- als auch Low-Level-APIs zum Hochladen großer Objekte. Diese APIs unterstützen verschlüsselungsbezogene Header in der Anforderung.

- Bei Verwendung der High-Level-`Transfer-Utility` -API stellen Sie wie im Folgenden dargestellt verschlüsselungsspezifische Header in der `TransferUtilityUploadRequest` bereit. Codebeispiele finden Sie unter Hochladen eines Objekts mit Multipart-Upload.

```
TransferUtilityUploadRequest request = new TransferUtilityUploadRequest()
{
```

```
    FilePath = filePath,
    BucketName = existingBucketName,
    Key = keyName,
    // Provide encryption information.
    ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
    ServerSideEncryptionCustomerProvidedKey = base64Key,
};
```

- Beim Verwenden der Low-Level-API geben Sie verschlüsselungsbezogene Informationen in der Anforderung zum Starten des mehrteiligen Uploads gefolgt von identischen Verschlüsselungs-Informationen in den nachfolgenden Anforderungen zum mehrteiligen Upload an. In der vollständigen Anforderung zum mehrteiligen Upload müssen Sie keine verschlüsselungsspezifischen Header angeben. Beispiele finden Sie unter Verwenden der - AWS SDKs (Low-Level-API).

  Nachfolgend finden Sie ein Beispiel für einen mehrteiligen Low-Level-Upload, der eine Kopie eines vorhandenen großen Objekts erstellt. Im Beispiel wird das zu kopierende Objekt in Amazon S3 mittels SSE-C gespeichert und Sie möchten das Zielobjekt ebenfalls mittels SSE-C speichern. Im Beispiel machen Sie folgendes:

  - Initiieren Sie eine mehrteilige Upload-Anfrage, indem Sie einen Verschlüsselungsschlüssel und zugehörige Informationen bereitstellen.
  - Stellen Sie Verschlüsselungsschlüssel für das Quell- und das Zielobjekt sowie die zugehörigen Informationen in der `CopyPartRequest` bereit.
  - Ermitteln Sie die Größe des zu kopierenden Quellobjekts, indem Sie die Objekt-Metadaten abrufen.
  - Laden Sie die Objekte in Teilen mit je 5 MB hoch.

  Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
```

```
{
    class SSECLowLevelMPUcopyObjectTest
    {
        private const string existingBucketName = "*** bucket name ***";
        private const string sourceKeyName      = "*** source object key name
 ***";
        private const string targetKeyName      = "*** key name for the target
 object ***";
        private const string filePath           = @"*** file path ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CopyObjClientEncryptionKeyAsync().Wait();
        }

        private static async Task CopyObjClientEncryptionKeyAsync()
        {
            Aes aesEncryption = Aes.Create();
            aesEncryption.KeySize = 256;
            aesEncryption.GenerateKey();
            string base64Key = Convert.ToBase64String(aesEncryption.Key);

            await CreateSampleObjUsingClientEncryptionKeyAsync(base64Key,
 s3Client);

            await CopyObjectAsync(s3Client, base64Key);
        }
        private static async Task CopyObjectAsync(IAmazonS3 s3Client, string
 base64Key)
        {
            List<CopyPartResponse> uploadResponses = new List<CopyPartResponse>();

            // 1. Initialize.
            InitiateMultipartUploadRequest initiateRequest = new
 InitiateMultipartUploadRequest
            {
                BucketName = existingBucketName,
                Key = targetKeyName,
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
```

```
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // 2. Upload Parts.
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB
            long firstByte = 0;
            long lastByte = partSize;

            try
            {
                // First find source object size. Because object is stored
encrypted with
                // customer provided key you need to provide encryption
information in your request.
                GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest()
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                    ServerSideEncryptionCustomerProvidedKey = base64Key // " *
**source object encryption key ***"
                };

                GetObjectMetadataResponse getObjectMetadataResponse = await
s3Client.GetObjectMetadataAsync(getObjectMetadataRequest);

                long filePosition = 0;
                for (int i = 1; filePosition <
getObjectMetadataResponse.ContentLength; i++)
                {
                    CopyPartRequest copyPartRequest = new CopyPartRequest
                    {
                        UploadId = initResponse.UploadId,
                        // Source.
                        SourceBucket = existingBucketName,
                        SourceKey = sourceKeyName,
                        // Source object is stored using SSE-C. Provide encryption
information.
```

```
                        CopySourceServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        CopySourceServerSideEncryptionCustomerProvidedKey =
base64Key, //"***source object encryption key ***",
                        FirstByte = firstByte,
                        // If the last part is smaller then our normal part size
then use the remaining size.
                        LastByte = lastByte >
getObjectMetadataResponse.ContentLength ?
                            getObjectMetadataResponse.ContentLength - 1 :
lastByte,

                        // Target.
                        DestinationBucket = existingBucketName,
                        DestinationKey = targetKeyName,
                        PartNumber = i,
                        // Encryption information for the target object.
                        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key
                    };
                    uploadResponses.Add(await
s3Client.CopyPartAsync(copyPartRequest));
                    filePosition += partSize;
                    firstByte += partSize;
                    lastByte += partSize;
                }

                // Step 3: complete.
                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = targetKeyName,
                    UploadId = initResponse.UploadId,
                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (Exception exception)
            {
                Console.WriteLine("Exception occurred: {0}", exception.Message);
```

```
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = targetKeyName,
                    UploadId = initResponse.UploadId
                };
                s3Client.AbortMultipartUpload(abortMPURequest);
            }
        }
        private static async Task
CreateSampleObjUsingClientEncryptionKeyAsync(string base64Key, IAmazonS3
s3Client)
        {
            // List to store upload part responses.
            List<UploadPartResponse> uploadResponses = new
List<UploadPartResponse>();

            // 1. Initialize.
            InitiateMultipartUploadRequest initiateRequest = new
InitiateMultipartUploadRequest
            {
                BucketName = existingBucketName,
                Key = sourceKeyName,
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };

            InitiateMultipartUploadResponse initResponse =
               await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // 2. Upload Parts.
            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                    {
                        BucketName = existingBucketName,
```

```
                        Key = sourceKeyName,
                        UploadId = initResponse.UploadId,
                        PartNumber = i,
                        PartSize = partSize,
                        FilePosition = filePosition,
                        FilePath = filePath,
                        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key
                    };

                    // Upload part and add response to our list.
                    uploadResponses.Add(await
s3Client.UploadPartAsync(uploadRequest));

                    filePosition += partSize;
                }

                // Step 3: complete.
                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId,
                    //PartETags = new List<PartETag>(uploadResponses)

                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);

            }
            catch (Exception exception)
            {
                Console.WriteLine("Exception occurred: {0}", exception.Message);
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId
                };
```

```
                await s3Client.AbortMultipartUploadAsync(abortMPURequest);
            }
        }
    }
}
```

Schützen von Daten mithilfe der clientseitigen Verschlüsselung

Clientseitige Verschlüsselung bezeichnet die lokale Verschlüsselung Ihrer Daten, um ihre Sicherheit bei der Übertragung und im Ruhezustand zu gewährleisten. Verwenden Sie den Amazon S3 Encryption Client, um Ihre Objekte zu verschlüsseln, bevor Sie sie an Amazon S3 senden. Wenn Ihre Objekte auf diese Weise verschlüsselt werden, werden Ihre Objekte nicht an Dritte weitergegeben, einschließlich AWS. Amazon S3 empfängt Ihre Objekte bereits verschlüsselt. Amazon S3 ist an der Verschlüsselung oder Entschlüsselung Ihrer Objekte nicht beteiligt. Sie können sowohl den Amazon S3 Encryption Client als auch die serverseitige Verschlüsselung verwenden, um Ihre Daten zu verschlüsseln. Wenn Sie verschlüsselte Objekte an Amazon S3 senden, erkennt Amazon S3 die Objekte nicht als verschlüsselt, sondern nur typische Objekte.

Der Amazon S3 Encryption Client fungiert als Vermittler zwischen Ihnen und Amazon S3. Nachdem Sie den Amazon S3 Encryption Client instanziiert haben, werden Ihre Objekte als Teil Ihrer `PutObject`- und `GetObject`-Anforderungen in Amazon S3 automatisch verschlüsselt. Ihre Objekte werden alle mit einem eindeutigen Datenschlüssel verschlüsselt. Der Amazon S3 Encryption Client verwendet keine Bucket-Schlüssel und interagiert nicht damit; dies gilt auch, wenn Sie einen KMS-Schlüssel als Umhüllungsschlüssel angeben.

Das Entwicklerhandbuch zum Amazon S3 Encryption Client konzentriert sich auf die Versionen 3.0 und höher des Amazon S3 Encryption Client. Weitere Informationen finden Sie unter Was ist der Amazon S3 Encryption Client? im Entwicklerhandbuch zum Amazon S3 Encryption Client.

Weitere Informationen zu früheren Versionen des Amazon S3-Verschlüsselungsclients finden Sie im AWS -SDK-Entwicklerhandbuch für Ihre Programmiersprache.

- AWS SDK for Java
- AWS SDK for .NET
- AWS SDK for Go
- AWS SDK for PHP
- AWS SDK for Ruby

- AWS SDK for C++

Richtlinie für den Datenverkehr zwischen Netzwerken

In diesem Thema wird beschrieben, wie Amazon S3 Verbindungen vom Service zu anderen Speicherorten sichert.

Datenverkehr zwischen Service und lokalen Clients und Anwendungen

Die folgenden Verbindungen können mit kombiniert werden AWS PrivateLink , um Konnektivität zwischen Ihrem privaten Netzwerk und bereitzustellen AWS:

- Eine AWS Site-to-Site-VPN-Verbindung. Weitere Informationen finden Sie unter Was ist AWS Site-to-Site VPN?
- Eine - AWS Direct Connect Verbindung. Weitere Informationen finden Sie unter Was ist AWS Direct Connect?

Der Zugriff auf Amazon S3 über das Netzwerk erfolgt über AWS veröffentlichte APIs. Clients müssen Transport Layer Security (TLS) 1.2 unterstützen. Wir empfehlen TLS 1.3. Clients müssen außerdem Cipher Suites mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi. Außerdem müssen Sie die Anfragen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signieren, die einem IAM-Prinzipal zugeordnet sind. Sie können auch AWS Security Token Service (STS) verwenden um temporäre Sicherheitsanmeldeinformationen zu generieren.

Datenverkehr zwischen AWS Ressourcen in derselben Region

Ein Virtual Private Cloud (VPC)-Endpunkt für Amazon S3 ist eine logische Einheit innerhalb einer VPC, die nur Konnektivität mit Amazon S3 ermöglicht. Die VPC leitet Anforderungen an Amazon S3 weiter und Antworten an die VPC zurück. Weitere Informationen finden Sie unter VPC-Endpunkte im Amazon VPC-Benutzerhandbuch. Dieser Abschnitt enthält Beispiele für Bucket-Richtlinien, die für die Steuerung des Zugriffs auf S3-Buckets von VPC-Endpunkten aus verwendet werden können. Siehe Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien.

AWS PrivateLink für Amazon S3

Mit AWS PrivateLink für Amazon S3 können Sie Schnittstellen-VPC-Endpunkte (Schnittstellenendpunkte) in Ihrer Virtual Private Cloud (VPC) bereitstellen. Diese Endpunkte sind direkt von Anwendungen aus zugänglich, die über VPN und On- AWS Direct Connect Premises oder in einem anderen AWS-Region über VPC-Peering verfügbar sind.

Schnittstellenendpunkte werden durch eine oder mehrere Elastic Network-Schnittstellen (ENIs) repräsentiert, denen private IP-Adressen aus Subnetzen in Ihrer VPC zugewiesen werden. Anforderungen an Amazon S3 über Schnittstellenendpunkte bleiben im Amazon-Netzwerk. Sie können auch von lokalen Anwendungen über AWS Direct Connect oder AWS Virtual Private Network () auf Schnittstellenendpunkte in Ihrer VPC zugreifen AWS VPN. Weitere Informationen darüber, wie Sie Ihre VPC mit Ihrem On-Premises-Netzwerk verbinden, finden Sie im AWS Direct Connect - Benutzerhandbuch und im AWS Site-to-Site VPN -Benutzerhandbuch.

Allgemeine Informationen zu Schnittstellen-Endpunkten finden Sie unter VPC-Schnittstellen-Endpunkte (AWS PrivateLink) im AWS PrivateLink -Handbuch.

Themen

Arten von VPC-Endpunkten für Amazon S3

Sie können zwei Arten von VPC-Endpunkten verwenden, um auf Amazon S3 zuzugreifen: Gateway-Endpunkte und Schnittstellenendpunkte (mit AWS PrivateLink). Ein Gateway-Endpunkt ist ein Gateway, das Sie in Ihrer Routing-Tabelle angeben, um von Ihrer VPC über das AWS Netzwerk auf Amazon S3 zuzugreifen. Schnittstellenendpunkte erweitern die Funktionalität von Gateway-Endpunkten, indem sie private IP-Adressen verwenden, um Anforderungen von Ihrer VPC, On-

Premises oder von einer VPC in einer anderen mithilfe AWS-Region von VPC-Peering oder an Amazon S3 weiterzuleiten AWS Transit Gateway. Weitere Informationen finden Sie unter Was ist VPC Peering? und Vergleich zwischen Transit Gateway und VPC-Peering.

Schnittstellenendpunkte sind mit Gateway-Endpunkten kompatibel. Wenn Sie einen vorhandenen Gateway-Endpunkt in der VPC haben, können Sie beide Arten von Endpunkten in derselben VPC verwenden.

Gateway-Endpunkte für Amazon S3	Schnittstellenendpunkte für Amazon S3
In beiden Fällen verbleibt Ihr Netzwerkverkehr im - AWS Netzwerk.	
Verwenden Sie öffentliche IP-Adressen von Amazon S3	Verwenden Sie private IP-Adressen aus Ihrer VPC für den Zugriff auf Amazon S3
Verwenden Sie die gleichen Amazon-S3-DNS-Namen	Erfordert endpunktspezifische Amazon-S3-DNS-Namen
Erlauben keinen On-Premises-Zugriff	Erlaubt On-Premises-Zugriff
Erlauben Sie keinen Zugriff von einem anderen AWS-Region	Erlauben des Zugriffs von einer VPC in einer anderen mithilfe AWS-Region von VPC-Peering oder AWS Transit Gateway
Nicht berechnet	Berechnet

Weitere Informationen zu Gateway-Endpunkten finden Sie unter Gateway-VPC-Endpunkte im AWS PrivateLink -Handbuch.

Einschränkungen und Einschränkungen von AWS PrivateLink für Amazon S3

VPC-Einschränkungen gelten AWS PrivateLink für für Amazon S3. Weitere Informationen finden Sie unter Überlegungen zu Schnittstellenendpunkten und AWS PrivateLink -Kontingente im Handbuch zu AWS PrivateLink . Darüber hinaus gelten die folgenden Einschränkungen.

AWS PrivateLink für Amazon S3 unterstützt Folgendes nicht:

- Endpunkte für den Federal Information Processing Standard (FIPS).

- Website-Endpunkte
- Globale Legacy-Endpunkte
- Endpunkte der S3-Dash-Region
- Amazon-S3-Dual-Stack-Endpunkte
- Verwenden von CopyObject oder UploadPartCopy zwischen Buckets in verschiedenen AWS-Regionen
- Transport Layer Security (TLS) 1.1

Erstellung eines VPC-Endpunkts

Informationen zum Erstellen eines VPC-Schnittstellenendpunkts finden Sie unter Erstellen eines VPC-Endpunkts im AWS PrivateLink -Handbuch.

Zugriff auf Amazon-S3-Schnittstellen-Endpunkte

Wenn Sie einen Schnittstellenendpunkt erstellen, generiert Amazon S3 zwei Arten von endpunktspezifischen S3-DNS-Namen: regional und zonengebunden.

- Ein regionaler DNS-Name enthält eine eindeutige VPC-Endpunkt-ID, eine Service-ID AWS-Region, die und `vpce.amazonaws.com` in seinem Namen. Beispielsweise könnte der generierte DNS-Name für VPC-Endpunkt-ID *`vpce-1a2b3c4d`* mit *`vpce-1a2b3c4d-5e6f`*`.s3.us-east-1.vpce.amazonaws.com` vergleichbar sein.
- Ein zonengebundener DNS-Name enthält die Availability Zone, z. B. *`vpce-1a2b3c4d-5e6f`***`-us-east-1a.`**`s3.us-east-1.vpce.amazonaws.com`. Sie können diese Option verwenden, wenn Ihre Architektur Availability Zones isoliert. Sie könnten sie beispielsweise zur Fehlereingrenzung oder zur Senkung der regionalen Datenübertragungskosten verwenden.

Endpunktspezifische S3-DNS-Namen können aus der öffentlichen S3-DNS-Domain aufgelöst werden.

Privates DNS

Optionen für private DNS für VPC-Schnittstellenendpunkte vereinfachen das Routing von S3-Datenverkehr über VPC-Endpunkte und helfen Ihnen dabei, den kostengünstigsten Netzwerkpfad für Ihre Anwendung zu nutzen. Sie können Optionen für private DNS verwenden, um regionalen S3-Datenverkehr weiterzuleiten, ohne Ihre S3-Clients aktualisieren zu müssen, damit sie die

endpunktspezifischen DNS-Namen Ihrer Schnittstellenendpunkte verwenden, und ohne die DNS-Infrastruktur verwalten zu müssen. Wenn private DNS-Namen aktiviert sind, werden regionale S3-DNS-Abfragen AWS PrivateLink für die folgenden Endpunkte in die privaten IP-Adressen von aufgelöst:

- Regionale Bucket-Endpunkte (z. B. `s3.us-east-1.amazonaws.com`)
- Kontrollendpunkte (z. B. `s3-control.us-east-1.amazonaws.com`)
- Zugriffspunkt-Endpunkte (z. B. `s3-accesspoint.us-east-1.amazonaws.com`)

Wenn Sie einen Gateway-Endpunkt in Ihrer VPC haben, können Sie In-VPC-Anforderungen automatisch über Ihren vorhandenen S3-Gateway-Endpunkt und On-Premises-Anforderungen über Ihren Schnittstellenendpunkt weiterleiten. Mit diesem Ansatz können Sie Ihre Netzwerkkosten optimieren, indem Sie Gateway-Endpunkte für Ihren In-VPC-Datenverkehr verwenden, die nicht in Rechnung gestellt werden. Ihre On-Premises-Anwendungen können AWS PrivateLink mithilfe des eingehenden Resolver-Endpunkts verwenden. Amazon stellt einen DNS-Server den Route 53 Resolver für Ihre VPC zur Verfügung. Ein eingehender Resolver-Endpunkt leitet DNS-Abfragen vom On-Premises Netzwerk an Route 53 Resolver weiter.

> **Important**
>
> Um bei Verwendung von Private DNS nur für eingehende Endpunkte aktivieren den kostengünstigsten Netzwerkpfad zu nutzen, muss in Ihrer VPC ein Gateway-Endpunkt vorhanden sein. Ein Gateway-Endpunkt hilft dabei, sicherzustellen, dass der In-VPC-Datenverkehr immer über das private AWS -Netzwerk weitergeleitet wird, wenn die Option Private DNS nur für eingehende Endpunkte aktivieren ausgewählt ist. Sie müssen diesen Gateway-Endpunkt beibehalten, solange die Option Private DNS nur für eingehende Endpunkte aktivieren aktiviert ist. Wenn Sie Ihren Gateway-Endpunkt löschen möchten, müssen Sie zuerst Private DNS nur für eingehende Endpunkte aktivieren deaktivieren. Wenn Sie einen vorhandenen Schnittstellenendpunkt auf Private DNS nur für eingehende Endpunkte aktivieren aktualisieren möchten, vergewissern Sie sich zunächst, dass Ihre VPC über einen S3-Gateway-Endpunkt verfügt. Weitere Informationen zu Gateway-Endpunkten und zur Verwaltung von privaten DNS-Namen finden Sie unter Gateway-VPC-Endpunkte bzw. DNS-Namen verwalten im Handbuch zu AWS PrivateLink .

Die Option Private DNS nur für eingehende Endpunkte aktivieren ist nur für Services verfügbar, die Gateway-Endpunkte unterstützen.

Weitere Informationen zum Erstellen eines VPC-Endpunkts, der Private DNS nur für eingehende Endpunkte aktivieren verwendet, finden Sie unter Erstellen eines Schnittstellenendpunkts im Handbuch zu AWS PrivateLink .

Verwenden der VPC-Konsole

In der Konsole haben Sie zwei Optionen: DNS-Namen aktivieren und Private DNS nur für eingehende Endpunkte aktivieren. DNS-Namen aktivieren ist eine Option, die von unterstützt wird AWS PrivateLink. Mit der Option DNS-Namen aktivieren können Sie die private Konnektivität von Amazon mit Amazon S3 nutzen und gleichzeitig Anforderungen an die standardmäßigen DNS-Namen für öffentliche Endpunkte ausführen. Wenn diese Option aktiviert ist, können Kunden den für Ihre Anwendung kostengünstigsten Netzwerkpfad nutzen.

Wenn Sie private DNS-Namen für einen vorhandenen oder neuen VPC-Schnittstellenendpunkt für Amazon S3 aktivieren, ist die Option Private DNS nur für eingehende Endpunkte aktivieren standardmäßig ausgewählt. Wenn diese Option ausgewählt ist, verwenden Ihre Anwendungen nur Schnittstellenendpunkte für Ihren On-Premises-Datenverkehr. Dieser In-VPC-Datenverkehr verwendet automatisch die kostengünstigeren Gateway-Endpunkte. Alternativ können Sie Private DNS nur für eingehende Endpunkte aktivieren deaktivieren, um alle S3-Anforderungen über Ihren Schnittstellenendpunkt weiterzuleiten.

Verwenden der AWS CLI

Wenn Sie keinen Wert für `PrivateDnsOnlyForInboundResolverEndpoint` angeben, wird standardmäßig `true` verwendet. Bevor Ihre VPC Ihre Einstellungen anwendet, überprüft sie jedoch, ob in der VPC ein Gateway-Endpunkt vorhanden ist. Wenn ein Gateway-Endpunkt in der VPC vorhanden ist, ist der Aufruf erfolgreich. Wenn nicht, wird die folgende Fehlermeldung angezeigt:

Um PrivateDnsOnlyForInboundResolverEndpoint auf „true“ zu setzen, muss die VPC *`vpce_id`* über einen Gateway-Endpunkt für den Service verfügen.

Für einen neuen VPC-Schnittstellenendpunkt

Verwenden Sie die Attribute `private-dns-enabled` und `dns-options`, um privates DNS über die Befehlszeile zu aktivieren. Die Option `PrivateDnsOnlyForInboundResolverEndpoint` im Attribut `dns-options` muss auf `true` gesetzt sein. Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws ec2 create-vpc-endpoint \
--region us-east-1 \
```

```
--service-name s3-service-name \
--vpc-id client-vpc-id \
--subnet-ids client-subnet-id \
--vpc-endpoint-type Interface  \
--private-dns-enabled  \
--ip-address-type ip-address-type \
--dns-options PrivateDnsOnlyForInboundResolverEndpoint=true \
--security-group-ids client-sg-id
```

Für einen vorhandenen VPC-Endpunkt

Wenn Sie privates DNS für einen vorhandenen VPC-Endpunkt verwenden möchten, verwenden Sie den folgenden Beispielbefehl und ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

```
aws ec2 modify-vpc-endpoint \
--region us-east-1 \
--vpc-endpoint-id client-vpc-id \
--private-dns-enabled \
--dns-options PrivateDnsOnlyForInboundResolverEndpoint=false
```

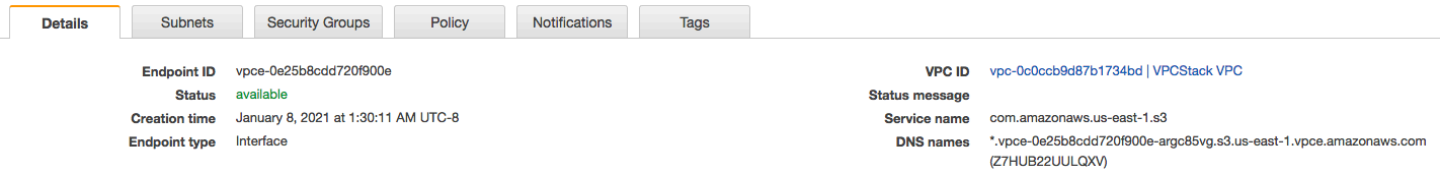
Wenn Sie einen vorhandenen VPC-Endpunkt aktualisieren möchten, um privates DNS nur für den eingehenden Resolver zu aktivieren, verwenden Sie das folgende Beispiel und ersetzen Sie die Beispielwerte durch Ihre eigenen.

```
aws ec2 modify-vpc-endpoint \
--region us-east-1 \
--vpc-endpoint-id client-vpc-id \
--private-dns-enabled \
--dns-options PrivateDnsOnlyForInboundResolverEndpoint=true
```

Zugriff auf Buckets, Zugriffspunkte und Amazon-S3-Control-API-Operationen über S3-Schnittstellenendpunkte

Sie können die AWS CLI oder AWS SDKs verwenden, um über S3-Schnittstellenendpunkte auf Buckets, S3-Zugriffspunkte und Amazon S3-Control-API-Operationen zuzugreifen.

Die folgende Abbildung zeigt den Tab Details der VPC-Konsole, auf der Sie den DNS-Namen eines VPC-Endpunkts finden. In diesem Beispiel lautet die VPC-Endpunkt-ID (vpce-id) `vpce-0e25b8cdd720f900e` und der DNS-Name lautet `*.vpce-0e25b8cdd720f900e-argc85vg.s3.us-east-1.vpce.amazonaws.com`.



Wenn Sie den DNS-Namen verwenden, um auf eine Ressource zuzugreifen, ersetzen Sie * durch den entsprechenden Wert. Die entsprechenden Werte, die anstelle von * verwendet werden können, lauten wie folgt:

- `bucket`
- `accesspoint`
- `control`

Um beispielsweise auf einen Bucket zuzugreifen, verwenden Sie einen DNS-Namen wie diesen:

`bucket.vpce-0e25b8cdd720f900e-argc85vg.s3.us-east-1.vpce.amazonaws.com`

Beispiele für die Verwendung von DNS-Namen für den Zugriff auf Buckets, Zugriffspunkte und Amazon-S3-Control API-Operationen finden Sie in den folgenden Abschnitten von AWS CLI Beispiele und AWS SDK-Beispiele.

Weitere Informationen zum Anzeigen Ihrer endpunktspezifischen DNS-Namen finden Sie unter Anzeigen der privaten DNS-Namenskonfiguration eines Endpunktservice im VPC-Benutzerhandbuch.

AWS CLI Beispiele

Verwenden Sie die `--endpoint-url` Parameter `--region` und , um über S3-Schnittstellenendpunkte in - AWS CLI Befehlen auf S3-Buckets, S3-Zugriffspunkte oder Amazon S3-Control-API-Operationen zuzugreifen.

Beispiel: Verwenden einer Endpunkt-URL zum Auflisten von Objekten in Ihrem Bucket

Ersetzen Sie im folgenden Beispiel den Bucket-Namen *my-bucket*, die Region *us-east-1* und den DNS-Namen der VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

```
aws s3 ls s3://my-bucket/ --region us-east-1 --endpoint-url
 https://bucket.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com
```

Beispiel: Verwenden einer Endpunkt-URL zum Auflisten von Objekten von einem Zugriffspunkt

- Methode 1 – Verwenden des Amazon-Ressourcennamens (ARN) des Zugriffspunkts mit dem Zugriffspunkt-Endpunkt

  Ersetzen Sie den ARN *us-east-1:123456789012:accesspoint/accesspointexamplename*, die Region *us-east-1* und die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

  ```
  aws s3api list-objects-v2 --bucket arn:aws:s3:us-east-1:123456789012:accesspoint/
  accesspointexamplename --region us-east-1 --endpoint-url
   https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com
  ```

  Wenn Sie den Befehl nicht erfolgreich ausführen können, aktualisieren Sie Ihre AWS CLI auf die neueste Version und versuchen Sie es erneut. Weitere Informationen zur Aktualisierung finden Sie unter Installation oder Aktualisierung der aktuellen Version der AWS CLI im Benutzerhandbuch zu AWS Command Line Interface .

- Methode 2 – Verwenden des Alias des Zugriffspunkts mit dem regionalen Bucket-Endpunkt

  Ersetzen Sie im folgenden Beispiel den Zugriffspunkt-Alias *accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias*, die Region *us-east-1* und die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

  ```
  aws s3api list-objects-v2 --
  bucket accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias
   --region us-east-1 --endpoint-url https://bucket.vpce-1a2b3c4d-5e6f.s3.us-
  east-1.vpce.amazonaws.com
  ```

- Methode 3 – Verwenden des Alias des Zugriffspunkts mit dem Zugriffspunkt-Endpunkt

Um einen S3-Endpunkt zu erstellen, bei dem der Bucket als Teil des Hostnamens enthalten ist, legen Sie zunächst den Adressierungsstil auf `virtual` fest, damit `aws s3api` ihn verwenden kann. Weitere Informationen zu `AWS configure` finden Sie unter Einstellungen der Konfigurations- und Anmeldeinformationsdatei im Benutzerhandbuch zu AWS Command Line Interface .

```
aws configure set default.s3.addressing_style virtual
```

Ersetzen Sie im folgenden Beispiel dann den Zugriffspunkt-Alias *accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias*, die Region *us-east-1* und die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen. Weitere Informationen zum Zugriffspunkt-Alias finden Sie unter Verwenden eines Alias im Bucket-Stil für Ihren S3-Bucket-Zugriffspunkt.

```
aws s3api list-objects-v2 --
bucket accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias --
region us-east-1 --endpoint-url https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com
```

Beispiel: Verwenden einer Endpunkt-URL zum Auflisten von Aufträgen mit einer S3-Control-API-Operation

Ersetzen Sie im folgenden Beispiel die Region *us-east-1*, die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* und die Konto-ID *12345678* durch Ihre eigenen Informationen.

```
aws s3control --region us-east-1 --endpoint-url
 https://control.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com list-jobs --
account-id 12345678
```

AWS SDK-Beispiele

Um über S3S3-Schnittstellenendpunkte auf S3-Buckets, S3-Zugriffspunkte oder Amazon S3-Control-API-Operationen zuzugreifen, wenn Sie die AWS SDKs verwenden, aktualisieren Sie Ihre SDKs auf die neueste Version. Konfigurieren Sie Ihre Clients anschließend so, dass sie eine Endpunkt-URL

verwenden, um über S3-Schnittstellenendpunkte auf einen Bucket, einen Zugriffspunkt oder Amazon-S3-Control-API-Operationen zuzugreifen.

SDK for Python (Boto3)

Beispiel: Verwenden einer Endpunkt-URL, um auf einen S3-Bucket zuzugreifen

Ersetzen Sie im folgenden Beispiel die Region *us-east-1* und die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

```
s3_client = session.client(
service_name='s3',
region_name='us-east-1',
endpoint_url='https://bucket.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com'
)
```

Beispiel: Verwenden einer Endpunkt-URL, um auf einen S3-Zugriffspunkt zuzugreifen

Ersetzen Sie im folgenden Beispiel die Region *us-east-1* und die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

```
ap_client = session.client(
service_name='s3',
region_name='us-east-1',
endpoint_url='https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com'
)
```

Beispiel: Verwenden einer Endpunkt-URL, um auf die Amazon-S3-Control-API zuzugreifen

Ersetzen Sie im folgenden Beispiel die Region *us-east-1* und die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

```
control_client = session.client(
service_name='s3control',
region_name='us-east-1',
endpoint_url='https://control.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com'
)
```

SDK for Java 1.x

Beispiel: Verwenden einer Endpunkt-URL, um auf einen S3-Bucket zuzugreifen

Ersetzen Sie im folgenden Beispiel die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

```
// bucket client
final AmazonS3 s3 = AmazonS3ClientBuilder.standard().withEndpointConfiguration(
        new AwsClientBuilder.EndpointConfiguration(
                "https://bucket.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com",
                Regions.DEFAULT_REGION.getName()
        )
).build();
List<Bucket> buckets = s3.listBuckets();
```

Beispiel: Verwenden einer Endpunkt-URL, um auf einen S3-Zugriffspunkt zuzugreifen

Ersetzen Sie im folgenden Beispiel die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* und den ARN *us-east-1:123456789012:accesspoint/prod* durch Ihre eigenen Informationen.

```
// accesspoint client
final AmazonS3 s3accesspoint =
 AmazonS3ClientBuilder.standard().withEndpointConfiguration(
        new AwsClientBuilder.EndpointConfiguration(
                "https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com",
                Regions.DEFAULT_REGION.getName()
        )
).build();
ObjectListing objects = s3accesspoint.listObjects("arn:aws:s3:us-
east-1:123456789012:accesspoint/prod");
```

Beispiel: Verwenden einer Endpunkt-URL, um auf eine Amazon-S3-Control-API-Operation zuzugreifen

Ersetzen Sie im folgenden Beispiel die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* durch Ihre eigenen Informationen.

```
// control client
```

```
final AWSS3Control s3control =
 AWSS3ControlClient.builder().withEndpointConfiguration(
        new AwsClientBuilder.EndpointConfiguration(
                "https://control.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com",
                Regions.DEFAULT_REGION.getName()
        )
).build();
final ListJobsResult jobs = s3control.listJobs(new ListJobsRequest());
```

SDK for Java 2.x

Beispiel: Verwenden einer Endpunkt-URL, um auf einen S3-Bucket zuzugreifen

Ersetzen Sie im folgenden Beispiel die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* und die Region *Region.US_EAST_1* durch Ihre eigenen Informationen.

```
// bucket client
Region region = Region.US_EAST_1;
s3Client = S3Client.builder().region(region)

 .endpointOverride(URI.create("https://bucket.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com"))
                   .build()
```

Beispiel: Verwenden einer Endpunkt-URL, um auf einen S3-Zugriffspunkt zuzugreifen

Ersetzen Sie im folgenden Beispiel die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* und die Region *Region.US_EAST_1* durch Ihre eigenen Informationen.

```
// accesspoint client
Region region = Region.US_EAST_1;
s3Client = S3Client.builder().region(region)

 .endpointOverride(URI.create("https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com"))
                   .build()
```

Beispiel: Verwenden einer Endpunkt-URL, um auf die Amazon-S3-Control-API zuzugreifen

Ersetzen Sie im folgenden Beispiel die VPC-Endpunkt-ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* und die Region *Region.US_EAST_1* durch Ihre eigenen Informationen.

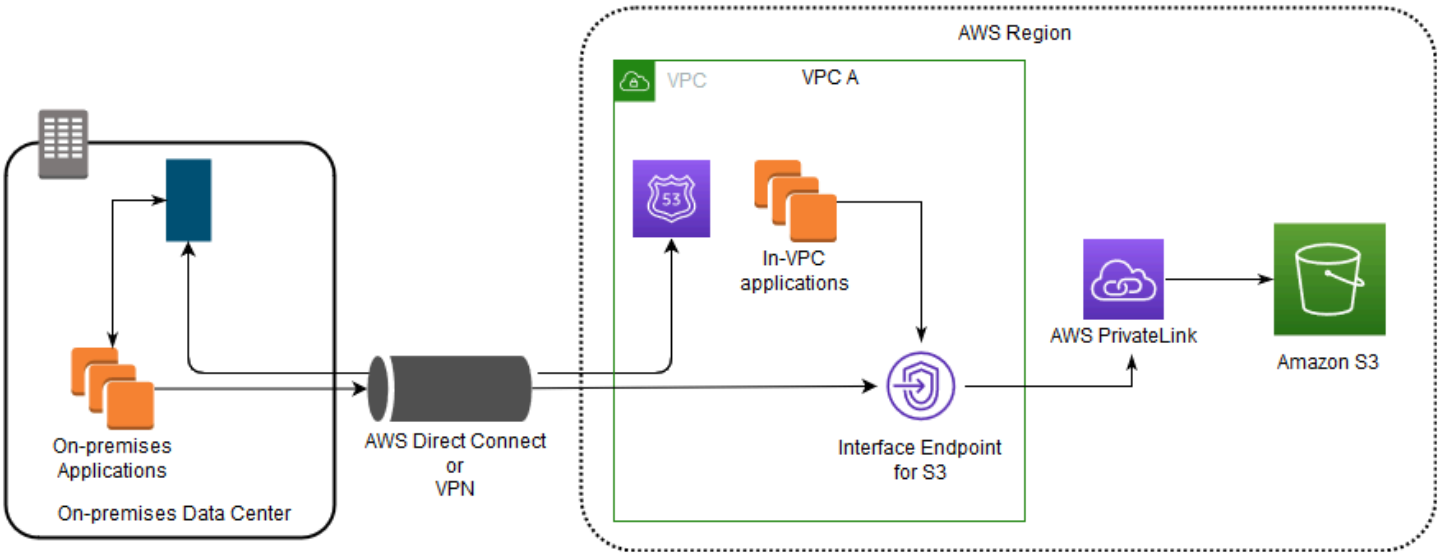
```
// control client
Region region = Region.US_EAST_1;
s3ControlClient = S3ControlClient.builder().region(region)

 .endpointOverride(URI.create("https://control.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com"))
                                  .build()
```

Aktualisieren einer lokalen DNS-Konfiguration

Wenn Sie endpunktspezifische DNS-Namen für den Zugriff auf die Schnittstellenendpunkte für Amazon S3 verwenden, brauchen Sie Ihren lokalen DNS-Resolver nicht zu aktualisieren. Sie können den endpunktspezifischen DNS-Namen mit der privaten IP-Adresse des Schnittstellenendpunkts aus der öffentlichen Amazon S3 DNS Domain auflösen.

Verwenden von Schnittstellenendpunkten für den Zugriff auf Amazon S3 ohne Gateway-Endpunkt oder Internet-Gateway in der VPC

Schnittstellenendpunkte in Ihrer VPC können sowohl In-VPC-Anwendungen als auch lokale Anwendungen über das Amazon-Netzwerk an Amazon S3 weiterleiten, wie im folgenden Diagramm dargestellt.

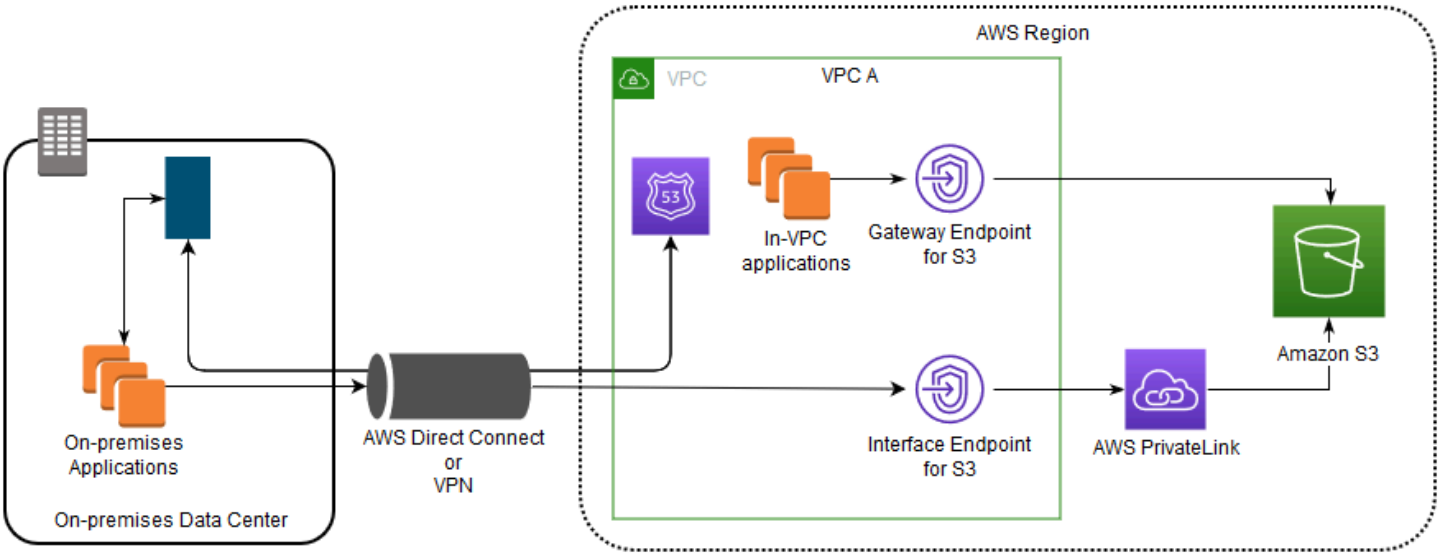


Das Diagramm veranschaulicht folgende Vorgänge:

- Ihr On-Premises-Netzwerk verwendet AWS Direct Connect oder AWS VPN , um eine Verbindung zu VPC A herzustellen.
- Ihre lokalen Anwendungen und Anwendungen in VPC A verwenden endpunktspezifische DNS-Namen, um über den S3-Schnittstellenendpunkt auf Amazon S3 zuzugreifen.
- On-Premises-Anwendungen senden Daten über AWS Direct Connect (oder AWS VPN) an den Schnittstellenendpunkt in der VPC. AWS PrivateLink verschiebt die Daten vom Schnittstellenendpunkt über das AWS Netzwerk zu Amazon S3.
- In-VPC-Anwendungen senden auch Datenverkehr an den Schnittstellenendpunkt. AWS PrivateLink verschiebt die Daten vom Schnittstellenendpunkt über das AWS Netzwerk zu Amazon S3.

Gemeinsames Verwenden von Gateway-Endpunkten und Schnittstellenendpunkten in derselben VPC für den Zugriff auf Amazon S3

Sie können Schnittstellenendpunkte erstellen und den vorhandenen Gateway-Endpunkt in derselben VPC beibehalten, wie das folgende Diagramm zeigt. Mit diesem Ansatz erlauben Sie In-VPC-Anwendungen, weiterhin über den Gateway-Endpunkt auf Amazon S3 zuzugreifen, was nicht in Rechnung gestellt wird. Dann würden nur Ihre On-Premises-Anwendungen Schnittstellenendpunkte für den Zugriff auf Amazon S3 verwenden. Um auf diese Weise auf Amazon S3 zuzugreifen, müssen Sie Ihre On-Premises-Anwendungen aktualisieren, damit endpunktspezifische DNS-Namen für Amazon S3 verwendet werden.



Das Diagramm veranschaulicht folgende Vorgänge:

- On-Premises-Anwendungen verwenden endpunktspezifische DNS-Namen, um Daten über AWS Direct Connect (oder AWS VPN) an den Schnittstellenendpunkt innerhalb der VPC zu senden. AWS PrivateLink verschiebt die Daten vom Schnittstellenendpunkt über das AWS Netzwerk an Amazon S3.
- Mithilfe von standardmäßigen regionalen Amazon S3-Namen senden In-VPC-Anwendungen Daten an den Gateway-Endpunkt, der über das AWS Netzwerk eine Verbindung zu Amazon S3 herstellt.

Weitere Informationen zu Gateway-Endpunkten finden Sie unter Gateway-VPC-Endpunkte im VPC-Benutzerhandbuch.

Erstellen einer VPC-Endpunktrichtlinie für Amazon S3

Sie können eine Endpunktrichtlinie an Ihren VPC-Endpunkt anhängen, der den Zugriff auf Amazon S3 steuert. Die Richtlinie gibt die folgenden Informationen an:

- Der AWS Identity and Access Management (IAM)-Prinzipal, der Aktionen ausführen kann
- Aktionen, die ausgeführt werden können
- Ressourcen, für die Aktionen ausgeführt werden können

Sie können Amazon-S3-Bucket-Richtlinien auch verwenden, um den Zugriff auf bestimmte Buckets von einem bestimmten VPC-Endpunkt aus zu beschränken, indem Sie die Bedingung `aws:sourceVpce` in Ihrer Bucket-Richtlinie verwenden. Die folgenden Beispiele zeigen Richtlinien, die den Zugriff auf einen Bucket oder einen Endpunkt einschränken.

Themen

- Beispiel: Beschränken des Zugriffs auf einen bestimmten Bucket von einem VPC-Endpunkt aus
- Beispiel: Beschränken des Zugriffs auf Buckets in einem bestimmten Konto von einem VPC-Endpunkt aus
- Beispiel: Beschränken des Zugriffs auf einen bestimmten VPC-Endpunkt in der S3-Bucket-Richtlinie

Beispiel: Beschränken des Zugriffs auf einen bestimmten Bucket von einem VPC-Endpunkt aus

Sie können eine Endpunktrichtlinie erstellen, die den Zugriff auf spezifische Amazon-S3-Buckets beschränkt. Diese Art von Richtlinie ist nützlich, wenn Sie andere AWS-Services in Ihrer VPC haben,

die Buckets verwenden. Die folgende Bucket-Richtlinie schränkt den Zugriff auf ausschließlich *DOC-EXAMPLE-BUCKET1* ein. Um diese Endpunktrichtlinie zu verwenden, ersetzen Sie *DOC-EXAMPLE-BUCKET1* im Beispiel durch den Namen Ihres Buckets.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909151",
  "Statement": [
    { "Sid": "Access-to-specific-bucket-only",
      "Principal": "*",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
                   "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"]
    }
  ]
}
```

Beispiel: Beschränken des Zugriffs auf Buckets in einem bestimmten Konto von einem VPC-Endpunkt aus

Sie können eine Endpunktrichtlinie erstellen, die den Zugriff auf die S3-Buckets in einer bestimmten einschränkt AWS-Konto. Verwenden Sie die folgende Anweisung in Ihrer Endpunktrichtlinie, um Clients in Ihrer VPC daran zu hindern, auf Buckets zuzugreifen, für die Sie keine Rechte haben. Die folgende Beispielanweisung erstellt eine Richtlinie, die den Zugriff auf Ressourcen einschränkt, die einer einzelnen AWS-Konto -ID, *111122223333*, gehören.

```
{
  "Statement": [
    {
      "Sid": "Access-to-bucket-in-specific-account-only",
      "Principal": "*",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:s3:::*",
```

```
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": "111122223333"
        }
      }
    }
  ]
}
```

> **Note**
>
> Um die AWS-Konto ID der Ressource anzugeben, auf die zugegriffen wird, können Sie entweder den - `aws:ResourceAccount` oder den -`s3:ResourceAccount`Schlüssel in Ihrer IAM-Richtlinie verwenden. Beachten Sie jedoch, dass einige auf den Zugriff auf AWS verwaltete Buckets AWS-Services angewiesen sind. Daher kann es auch den Zugriff auf diese Ressourcen beeinflussen, wenn Sie den Schlüssel `aws:ResourceAccount` oder `s3:ResourceAccount` in Ihrer IAM-Richtlinie verwenden.

Beispiel: Beschränken des Zugriffs auf einen bestimmten VPC-Endpunkt in der S3-Bucket-Richtlinie

Beispiel: Beschränken des Zugriffs auf einen bestimmten VPC-Endpunkt in der S3-Bucket-Richtlinie

Die folgende Amazon-S3-Bucket-Richtlinie ermöglicht den Zugriff auf einen spezifischen Bucket, *DOC-EXAMPLE-BUCKET2*, nur vom VPC-Endpunkt *vpce-1a2b3c4d* aus. Die Richtlinie lehnt sämtlichen Zugriff auf den Bucket ab, der nicht über den angegebenen Endpunkt erfolgt.
Die Bedingung `aws:sourceVpce` gibt den Endpunkt an und erfordert keinen Amazon-Ressourcennamen (ARN) für die VPC-Endpunkt-Ressource, sondern nur die Endpunkt-ID. Um diese Bucket-Richtlinie zu verwenden, ersetzen Sie *DOC-EXAMPLE-BUCKET2* und *vpce-1a2b3c4d* durch den Namen Ihres Buckets und Ihren Endpunkt.

> **Important**
>
> - Wenn Sie die folgende Amazon-S3-Bucket-Richtlinie anwenden, um den Zugriff auf bestimmte VPC-Endpunkte einzuschränken, blockieren Sie möglicherweise unbeabsichtigt Ihren Zugriff auf den Bucket. Bucket-Richtlinien, die den Bucket-Zugriff auf Verbindungen einschränken sollen, die von Ihrem VPC-Endpunkt ausgehen, können alle Verbindungen

zum Bucket blockieren. Informationen zur Behebung dieses Problems finden Sie unter My bucket policy has the wrong VPC or VPC endpoint ID (Meine Bucket-Richtlinie hat die falsche VPC- oder VPC-Endpunkt-ID). Wie kann ich die Richtlinie so ändern, dass ich auf den Bucket zugreifen kann? im AWS Support Knowledge Center.

- Bevor Sie die folgende Beispielrichtlinie verwenden, ersetzen Sie die VPC-Endpunkt-ID durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls können Sie nicht auf Ihren Bucket zugreifen.
- Diese Richtlinie deaktiviert den Konsolenzugriff auf den angegebenen Bucket, da Konsolenanforderungen nicht vom angegebenen VPC-Endpunkt stammen.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909152",
  "Statement": [
    { "Sid": "Access-to-specific-VPCE-only",
      "Principal": "*",
      "Action": "s3:*",
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET2",
                   "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*"],
      "Condition": {"StringNotEquals": {"aws:sourceVpce": "vpce-1a2b3c4d"}}
    }
  ]
}
```

Weitere Beispiele für Richtlinien finden Sie in Endpunkte für Amazon S3 im VPC-Benutzerhandbuch.

Weitere Informationen zur VPC-Konnektivität finden Sie unter Konnektivitätsoptionen von Netzwerk zu VPC im AWS Whitepaper Amazon Virtual Private Cloud Connectivity Options.

Identity and Access Management in Amazon S3

Standardmäßig sind alle Amazon-S3-Ressourcen – Buckets, Objekte und zugehörige Unterressourcen (z. B. `lifecycle` Konfiguration und `website` Konfiguration) – privat. Nur der Ressourcenbesitzer, das , das es erstellt AWS-Konto hat, kann auf die Ressource zugreifen. Der Ressourcenbesitzer kann optional anderen Zugriffsberechtigungen erteilen, indem er eine Zugriffsrichtlinie schreibt.

Amazon S3 unterstützt Optionen für Zugriffsrichtlinien, die ganz allgemein als ressourcenbasierte Richtlinien und Benutzerrichtlinien unterteilt werden können. Zugriffsrichtlinien, die Sie Ihren Ressourcen hinzufügen (Buckets und Ordnern) werden als ressourcenbasierte Richtlinien bezeichnet. Beispielsweise sind Bucket-Richtlinien und Zugriffspunkt-Richtlinien ressourcenbasierte Richtlinien. Sie können auch Benutzern in Ihrem Konto Richtlinien hinzufügen. Diese werden als Benutzerrichtlinien bezeichnet. Sie können ressourcenbasierte Richtlinien, Benutzerrichtlinien oder eine Kombination daraus verwenden, um Ihre Berechtigungen für Ihre Amazon-S3-Ressourcen zu verwalten. Sie können auch Zugriffskontrolllisten (ACLs) verwenden, um anderen AWS-Konten grundlegende Lese- und Schreibberechtigungen zu erteilen.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden)

Weitere Informationen zu den häufigsten Ursachen für den Fehler „Zugriff verweigert“ (403 Forbidden) in Amazon S3 finden Sie unter Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3.

Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3

Eine vollständige Liste der IAM-Berechtigungen, -Ressourcen und -Bedingungsschlüssel für Amazon S3 finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Serviceautorisierungsreferenz.

Weitere Informationen

Weitere Informationen zur Verwaltung des Zugriffs auf Ihre Amazon-S3-Objekte und -Buckets finden Sie in den folgenden Themen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Übersicht über die Verwaltung von Zugriffsberechtigungen

Beim Erteilen von Berechtigungen in Amazon S3 entscheiden Sie, wer die Berechtigungen erhält, für welche Amazon-S3-Ressourcen die Berechtigungen gelten und welche Aktionen zu diesen Ressourcen gestattet werden sollen. Die folgenden Abschnitte geben einen Überblick über Amazon-S3-Ressourcen und wie Sie die beste Methode zur Kontrolle des Zugriffs auf diese ermitteln können.

Themen

Amazon-S3-Ressourcen: Buckets und Objekte

In ist AWS eine Ressource eine Entität, mit der Sie arbeiten können. In Amazon S3 sind Buckets und Objekte die Ressourcen und beiden sind Subressourcen zugeordnet.

Zu den Bucket-Subressourcen gehören unter anderem:

- `lifecycle` – speichert Lebenszyklus-Konfigurationsinformationen. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.
- `website` – speichert Website-Konfigurationsinformationen, wenn Sie Ihren Bucket für das Website-Hosting konfigurieren. Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.
- `versioning` — speichert die Versionierungskonfiguration. Weitere Informationen finden Sie unter PUT-Bucket-Versionierung in Amazon Simple Storage Service – API-Referenz.
- `policy` und `acl` (Access Control List) – speichert Zugriffsberechtigungsinformationen für den Bucket.
- `cors` (Cross-Origin Resource Sharing) – unterstützt die Konfiguration Ihres Buckets so, dass ursprungsübergreifende Anforderungen möglich sind. Weitere Informationen finden Sie unter Cross-Origin Resource Sharing (CORS) verwenden.
- `object ownership` – ermöglicht dem Bucket-Eigentümer, neue Objekte im Bucket in Besitz zu nehmen, unabhängig davon, wer sie hochlädt. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..
- `logging` – ermöglicht Ihnen, Amazon S3 aufzufordern, Bucket-Zugriffsprotokolle zu speichern.

Zu den Objekt-Subressourcen gehören unter anderem:

- `acl` – Speichert eine Liste von Zugriffsberechtigungen für das Objekt. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.
- `restore` — unterstützt die temporäre Wiederherstellung eines archivierten Objekts. Weitere Informationen finden Sie unter POST Object restore in der API-Referenz zum Amazon Simple Storage Service.

  Ein Objekt in der Speicherklasse S3 Glacier Flexible Retrieval ist ein archiviertes Objekt. Für den Zugriff auf das Objekt müssen Sie zuerst eine Anforderung zur Wiederherstellung initiieren, die eine Kopie des archivierten Objekts wiederherstellt. Geben Sie in der Anforderung die Anzahl der

Tage an, wie lang die wiederhergestellte Kopie existieren soll. Weitere Informationen über das Archivieren von Objekten finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Amazon-S3-Bucket- und Objekt-Eigentümerschaft

Buckets und Objekte sind Amazon-S3-Ressourcen. Standardmäßig kann nur der jeweilige Eigentümer auf diese Ressourcen zugreifen. Der Ressourceneigentümer bezieht sich auf die AWS-Konto , die die Ressource erstellt. Beispielsweise:

- Das AWS-Konto , mit dem Sie Buckets erstellen und Objekte hochladen, ist Eigentümer dieser Ressourcen.
- Wenn Sie ein Objekt mit AWS Identity and Access Management (IAM)-Benutzer- oder -Rollenanmeldeinformationen hochladen, ist das , zu dem der Benutzer oder die Rolle gehört AWS-Konto , Eigentümer des Objekts.
- Ein Bucket-Eigentümer kann einem anderen AWS-Konto (oder Benutzern in einem anderen Konto) kontoübergreifende Berechtigungen zum Hochladen von Objekten erteilen. In diesem Fall gehören diese Objekte dem , AWS-Konto das Objekte hochlädt. Der Bucket-Eigentümer besitzt keine Berechtigungen für die Objekte, die anderen Konten gehören, mit den folgenden Ausnahmen:
  - Der Bucket-Eigentümer zahlt die Rechnungen. Ein Bucket-Eigentümer kann explizit den Zugriff auf Objekte verweigern oder Objekte im Bucket löschen, unabhängig davon, wem sie gehören.
  - Ein Bucket-Eigentümer kann Objekte archivieren oder archivierte Objekte wiederherstellen, unabhängig davon, wem sie gehören. Die Archivierung bezieht sich auf die Speicherklasse, die beim Speichern der Objekte verwendet wurde. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Eigentümerschaft und Anforderungsauthentifizierung

Alle Anforderungen nach einem Bucket sind authentifiziert oder nicht authentifiziert. Authentifizierte Anforderungen müssen einen Signaturwert enthalten, der den Absender der Anforderung authentifiziert, und für nicht authentifizierte Anforderungen gilt dies nicht. Weitere Informationen über Anforderungsauthentifizierung finden Sie unter Senden von Anforderungen.

Der Eigentümer eines Buckets kann nicht authentifizierte Anforderungen zulassen. So sind beispielsweise nicht authentifizierte PUT Object-Anforderungen erlaubt, wenn für einen Bucket eine öffentliche Bucket-Richtlinie gilt oder wenn ein Bucket-ACL `WRITE`- oder `FULL_CONTROL`-Zugriff für die Gruppe „All Users (Alle Benutzer)“ oder für anonyme Benutzer gewährt. Weitere Informationen

zu öffentlichen Bucket-Richtlinien und öffentlichen Zugriffskontrolllisten (ACLs) finden Sie unter Die Bedeutung von „öffentlich“.

Alle nicht authentifizierten Anforderungen werden vom anonymen Benutzer erstellt. Dieser Benutzer ist in ACLs durch die spezifische kanonische Benutzer-ID `65a011a29cdf8ec533ec3d1ccaae921c` repräsentiert. Wenn ein Objekt mit einer nicht authentifizierten Anforderung zu einem Bucket hochgeladen wird, ist der anonyme Benutzer Eigentümer des Objekts. Die Standard-Objekt-ACL gewährt dem anonymen Benutzer als dem Eigentümer des Objekts `FULL_CONTROL`. Daher erlaubt Amazon S3, dass nicht authentifizierte Anforderungen das Objekt abrufen oder seine ACL modifizieren.

Um zu verhindern, dass Objekte von dem anonymen Benutzer modifiziert werden, empfehlen wir, keine Bucketrichtlinien zu implementieren, die anonyme öffentliche Schreibvorgänge für Ihren Bucket erlauben, oder die ACLs verwenden, den dem anonymen Benutzer Schreibzugriff auf Ihren Bucket gewähren. Sie können diese empfohlene Verhaltensweise durch die Verwendung von Amazon S3 Block Public Access erzwingen.

Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht.

> **Important**
>
> Wir empfehlen, die Anmeldeinformationen des AWS-Konto Root-Benutzers nicht für authentifizierte Anforderungen zu verwenden. Erstellen Sie stattdessen eine IAM-Rolle, der Sie vollständigen Zugriff gewähren. Wir bezeichnen Benutzer mit dieser Rolle als Administratorbenutzer. Sie können anstelle von AWS-Konto Root-Benutzer-Anmeldeinformationen, die der Administratorrolle zugewiesen sind, Anmeldeinformationen verwenden, um mit zu interagieren AWS und Aufgaben auszuführen, z. B. um einen Bucket zu erstellen, Benutzer zu erstellen und Berechtigungen zu erteilen. Weitere Informationen finden Sie unter Stammbenutzer-Anmeldeinformationen von AWS-Konto und IAM-Benutzer-Anmeldeinformationen in der Allgemeine AWS-Referenz und unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Ressourcenvorgänge

Amazon S3 bietet eine Reihe von Vorgängen, um mit den Amazon-S3-Ressourcen zu arbeiten. Eine Liste der verfügbaren Operationen finden Sie unter Von Amazon S3 definierte Aktionen in der Service-Autorisierungs-Referenz.

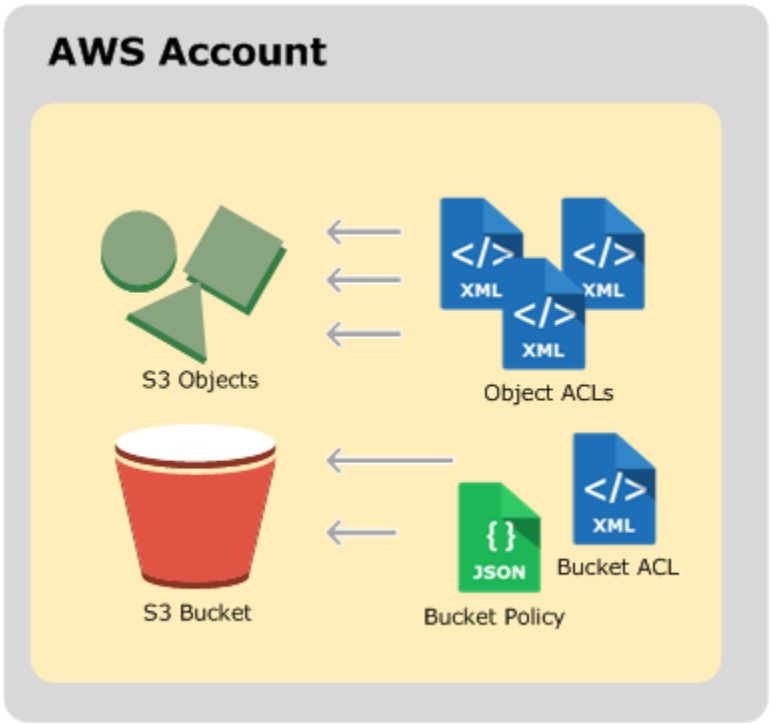
Verwalten des Zugriffs auf Ressourcen

Die Zugriffsverwaltung bezieht sich darauf, anderen (AWS-Konten und Benutzern) die Berechtigung zum Ausführen der Ressourcenoperationen durch Schreiben einer Zugriffsrichtlinie zu erteilen. Sie können beispielsweise einem Benutzer in einem die `PUT Object` Berechtigung erteilen, AWS-Konto damit der Benutzer Objekte in Ihren Bucket hochladen kann. Zusätzlich zur Erteilung von Berechtigungen für einzelne Benutzer und Konten können Sie jedem Berechtigungen erteilen (auch als anonymer Zugriff bezeichnet) oder allen authentifizierten Benutzern (Benutzern mit - AWS Anmeldeinformationen). Wenn Sie beispielsweise Ihren Bucket als Website konfigurieren, könnten Sie Objekt öffentlich machen, indem Sie jedem die `GET Object`-Berechtigung erteilen.

Zugriffsrichtlinienoptionen

Eine Zugriffsrichtlinie beschreibt, wer Zugriff auf welche Objekte hat. Sie können eine Zugriffsrichtlinie einer Ressource (Bucket und Objekt) oder einem Benutzer zuordnen. Dementsprechend können Sie die verfügbaren Amazon-S3-Zugriffsrichtlinien wie folgt kategorisieren:

- Auf Ressourcen basierende Richtlinien – Bucket-Richtlinien und Zugriffskontrolllisten (ACLs) sind auf Ressourcen basierend, weil Sie sie Ihren Amazon-S3-Ressourcen zuordnen.



- ACL – Jedem Bucket und Objekt ist eine Zugriffskontrollliste zugeordnet. Eine ACL listet die erteilten Berechtigungen auf, die den Berechtigungsempfänger und die erteilte Berechtigung identifizieren. Sie können ACLs verwenden, um anderen AWS-Konten grundlegende Lese-/Schreibberechtigungen zu erteilen. ACLs verwenden ein für Amazon S3 spezifisches XML-Schema.

  Es folgt ein Beispiel für eine Bucket-ACL. Das Recht der ACL zeigt einen Bucket-Eigentümer, der volle Kontrolle besitzt.

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>*** Owner-Canonical-User-ID ***</ID>
    <DisplayName>owner-display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>*** Owner-Canonical-User-ID ***</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
```

```
  </AccessControlList>
</AccessControlPolicy>
```

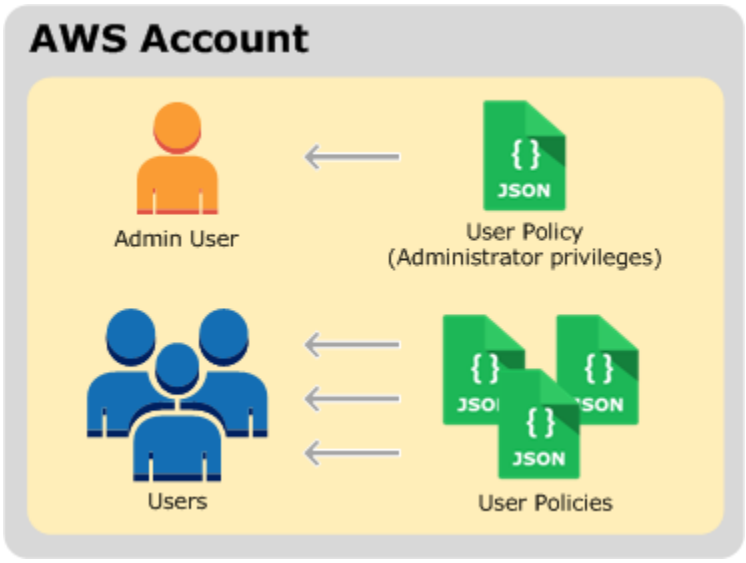
Bucket- und Objekt-ACLs verwenden dasselbe XML-Schema.

- Bucket-Richtlinie – Für Ihren Bucket können Sie eine Bucket-Richtlinie hinzufügen, um anderen AWS-Konten oder IAM-Benutzern Berechtigungen für den Bucket und die darin enthaltenen Objekte zu erteilen. Objektberechtigungen gelten nur für die Objekte, die der Bucket-Eigentümer erstellt. Bucket-Richtlinien ergänzen ACL-basierte Zugriffsrichtlinien, und in vielen Fällen ersetzen sie sie.

  Hier finden Sie ein Beispiel für eine Bucket-Richtlinie. Bucket-Richtlinien (und Benutzerrichtlinien) werden unter Verwendung einer JSON-Datei dargestellt. Die Richtlinie gewährt anonyme Berechtigung zum Lesen aller Objekte in einem Bucket. Die Bucket-Richtlinie enthält eine Anweisung, die die `s3:GetObject`-Aktion (Leseberechtigung) für Buckets im Bucket `examplebucket` erlaubt. Durch die Angabe von `principal` mit einem Platzhalterzeichen (*) erteilt die Richtlinie anonymen Zugriff und ist mit Vorsicht zu verwenden. Die folgende Bucket-Richtlinie würde beispielsweise Objekte öffentlich zugänglich machen.

  ```
  {
      "Version":"2012-10-17",
      "Statement": [
          {
              "Sid":"GrantAnonymousReadPermissions",
              "Effect":"Allow",
              "Principal": "*",
              "Action":["s3:GetObject"],
              "Resource":["arn:aws:s3:::awsexamplebucket1/*"]
          }
      ]
  }
  ```

- Benutzerrichtlinien – Sie können IAM für die Verwaltung des Zugriffs auf Ihre Amazon-S3-Ressourcen verwenden. Sie können IAM-Benutzer, -Gruppen und -Rollen in Ihrem -Konto erstellen und ihnen Zugriffsrichtlinien anfügen, die ihnen Zugriff auf - AWS Ressourcen gewähren, einschließlich Amazon S3.



Weitere Informationen über IAM finden Sie unter AWS Identity and Access Management (IAM).

Nachstehend finden Sie ein Beispiel für eine Benutzerrichtlinie. In einer IAM-Benutzerrichtlinie können Sie keine anonymen Berechtigungen gewähren, weil die Richtlinie mit einem Benutzer verknüpft ist. Die Beispielrichtlinie gestattet dem entsprechenden Benutzer, dass er sechs verschiedene Amazon-S3-Aktionen für einen Bucket und die darin enthaltenen Objekte ausführt. Sie können diese Richtlinie einem spezifischen IAM-Benutzer, einer Gruppe oder einer Rolle zuordnen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssignUserActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                 "arn:aws:s3:::awsexamplebucket1/*",
                 "arn:aws:s3:::awsexamplebucket1"
            ]
        },
```

```
        {
            "Sid": "ExampleStatement2",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        }
    ]
}
```

Wenn Amazon S3 eine Anfrage erhält, muss es alle Zugriffsrichtlinien auswerten, um festzustellen, ob es die Anfrage genehmigen oder verweigern soll. Weitere Informationen dazu, wie Amazon S3 diese Richtlinien auswertet, finden Sie unter Wie Amazon S3 eine Anforderung autorisiert.

IAM Access Analyzer für S3

In der Amazon-S3-Konsole können Sie IAM Access Analyzer für S3 verwenden, um alle Buckets zu überprüfen, die über Bucket-Zugriffssteuerungslisten (ACLs), Bucket-Richtlinien oder Zugriffspunktrichtlinien verfügen, die öffentlichen oder gemeinsamen Zugriff gewähren. IAM Access Analyzer für S3 macht Sie auf Buckets aufmerksam, die so konfiguriert sind, dass jedem im Internet oder anderen , einschließlich AWS-Konten außerhalb Ihrer Organisation AWS-Konten, Zugriff gewährt wird. Für jeden öffentlichen oder freigegebenen Bucket erhalten Sie Ergebnisse, die die Quelle und die Ebene des öffentlichen oder freigegebenen Zugriffs melden.

In IAM Access Analyzer für S3 können Sie den gesamten öffentlichen Zugriff auf einen Bucket mit einem einzigen Klick blockieren. Wir empfehlen Ihnen, den gesamten Zugriff auf Ihre Buckets zu blockieren, es sei denn, Sie benötigen öffentlichen Zugriff, um einen bestimmten Anwendungsfall zu unterstützen. Bevor Sie den gesamten öffentlichen Zugriff blockieren, stellen Sie sicher, dass Ihre Anwendungen ohne öffentlichen Zugriff weiterhin ordnungsgemäß funktionieren. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Sie können auch einen Drilldown in die Berechtigungseinstellungen auf Bucket-Ebene ausführen, um detaillierte Zugriffsebenen zu konfigurieren. Für bestimmte und geprüfte Anwendungsfälle, die öffentlichen oder freigegebenen Zugriff erfordern, können Sie Ihre Absicht bestätigen und aufzeichnen, dass der Bucket öffentlich oder freigegeben bleibt, indem Sie die Ergebnisse für den Bucket archivieren. Diese Bucket-Konfigurationen sind jederzeit aufrufbar und änderbar. Sie können Ihre Ergebnisse auch als CSV-Bericht zu Auditing-Zwecken herunterladen.

IAM Access Analyzer für S3 ist ohne zusätzliche Kosten in der Amazon-S3-Konsole verfügbar. IAM Access Analyzer für S3 wird von AWS Identity and Access Management (IAM) IAM Access Analyzer

unterstützt. Um IAM Access Analyzer für S3 auf der Amazon S3-Konsole zu verwenden, müssen Sie die IAM-Konsole aufrufen und einen Analysator auf Kontoebene in IAM Access Analyzer pro Region erstellen.

Weitere Informationen zu IAM Access Analyzer für S3 finden Sie unter Überprüfen des Bucket-Zugriffs mit IAM Access Analyzer für S3.

Welche Zugriffskontrollmethode sollte ich verwenden?

Angesichts der verschiedenen Optionen zum Schreiben einer Zugriffsrichtlinie ergeben sich die folgenden Fragen:

- Wann sollte ich welche Zugriffskontrollmethode verwenden? Sollte ich beispielsweise für das Erteilen von Bucket-Berechtigungen eine Bucket-Richtlinie oder eine Bucket-ACL verwenden?

  Ich besitze einen Bucket und die im Bucket enthaltenen Objekte. Sollte ich eine auf Ressourcen basierende Zugriffsrichtlinie oder eine auf IAM-Identitäten Benutzerrichtlinie verwenden?

  Wenn ich eine auf Ressourcen basierende Zugriffsrichtlinie verwende, sollte ich eine Bucket-Richtlinie oder eine Objekt-ACL verwenden, um Objektberechtigungen zu verwalten?
- Ich besitze einen Bucket, aber nicht alle darin enthaltenen Objekte. Wie werden Zugriffsberechtigungen für die Objekte verwaltet, die jemandem anderen gehören?
- Wenn ich den Zugriff über eine Kombination aus diesen Zugriffsrichtlinienoptionen erteile, wie kann Amazon S3 feststellen, ob ein Benutzer berechtigt ist, eine angefragte Operation auszuführen?

Die folgenden Abschnitte erklären diese Zugriffskontrollalternativen, wie Amazon S3 einen Zugriffskontrollmechanismus auswertet und wann welche Zugriffskontrollmethode zu verwenden ist. Sie bieten auch Beispiel-Anleitungen.

- Richtlinien für Zugriffsrichtlinien
- Wie Amazon S3 eine Anforderung autorisiert
- Beispiel-Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon-S3-Ressourcen
- Bewährte Methoden für die Zugriffssteuerung

Richtlinien für Zugriffsrichtlinien

Amazon S3 unterstützt ressourcenbasierte Richtlinien und Benutzerrichtlinien für die Verwaltung des Zugriffs auf Ihre Amazon-S3-Ressourcen. Weitere Informationen finden Sie unter Verwalten

des Zugriffs auf Ressourcen. Ressourcenbasierte Richtlinien umfassen Bucket-Richtlinien, Bucket-Zugriffskontrolllisten (ACLs) und Objekt-ACLs. Dieser Abschnitt beschreibt spezifische Szenarien für die Verwendung ressourcenbasierter Zugriffsrichtlinien für die Verwaltung des Zugriffs auf Ihre Amazon-S3-Ressourcen.

Themen

Wann eine ACL-basierte Zugriffsrichtlinie verwendet wird (Bucket- und Objekt-ACLs)

Buckets und Objekten sind ACLs zugeordnet, die Sie verwenden können, um Berechtigungen zu erteilen.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien verwenden, um Zugriff auf Ihren Bucket und die darin enthaltenen Objekte zu gewähren. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den

Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Wann Objekt-ACLs verwendet werden

Nachfolgend finden Sie die Szenarien, in denen Sie Objekt-ACLs zur Verwaltung von Berechtigungen verwenden sollten.

Objekte sind nicht im Besitz des Bucket-Eigentümers

Eine Objekt-ACL ist die einzige Möglichkeit, den Zugriff auf Objekte zu verwalten, die nicht dem Bucket-Eigentümer gehören. Ein AWS-Konto , dem der Bucket gehört, kann einem anderen die AWS-Konto Berechtigung erteilen, Objekte hochzuladen. Dem Bucket-Eigentümer gehören diese Objekte nicht. Das AWS-Konto , das das Objekt erstellt hat, muss Berechtigungen mithilfe von Objekt-ACLs erteilen.

> **Note**
>
> Ein Bucket-Eigentümer kann keine Berechtigungen für Objekte erteilen, die ihm nicht gehören. Beispielsweise gilt eine Bucket-Richtlinie, die Objektberechtigungen erteilt, nur für Objekte, die dem Bucket-Eigentümer gehören. Ein Bucket-Eigentümer, der die Rechnung zahlt, kann jedoch eine Bucket-Richtlinie schreiben, die den Zugriff auf Objekte im Bucket verweigern, unabhängig davon, wem dieser gehört. Der Bucket-Eigentümer kann auch beliebige Objekte im Bucket löschen

Sie müssen Berechtigungen auf Objektebene verwalten

Angenommen, die Berechtigungen variieren je nach Objekt und Sie müssen Berechtigungen auf Objektebene verwalten. Sie können eine einzelne Richtlinienanweisung schreiben, die einem AWS-Konto -Leseberechtigung für Millionen von Objekten mit einem bestimmten Schlüsselnamen-Präfix erteilt. Beispielsweise könnten Sie Leseberechtigung für Objekte erteilen, die mit dem Schlüsselnamepräfix `logs` beginnen. Wenn Ihre Zugriffsberechtigungen jedoch zwischen den Objekten variieren, ist die Erteilung von Berechtigungen für einzelne Objekte unter Verwendung einer Bucket-Richtlinie möglicherweise nicht praktisch. Außerdem sind Bucket-Richtlinien auf eine Größe von 20 KB beschränkt.

In diesem Fall ist die Verwendung von Objekt-ACLs möglicherweise eine geeignetere Alternative. Das gilt auch, obwohl eine Objekt-ACL ebenfalls auf maximal 100 Rechte beschränkt ist. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.

Objekt-ACLs steuern nur Berechtigungen auf Objektebene

Es gibt eine Bucket-Richtlinie für den gesamten Bucket, aber Objekt-ACLs werden pro Objekt eingerichtet.

Ein AWS-Konto , dem ein Bucket gehört, kann einem anderen die AWS-Konto Berechtigung erteilen, eine Zugriffsrichtlinie zu verwalten. Es gestattet diesem Konto, die Richtlinie beliebig zu verändern. Um die Berechtigungen besser verwalten zu können, sollten Sie keine so allgemeine Berechtigung erteilen, sondern besser nur `READ-ACP` und `WRITE-ACP`-Berechtigungen für Untermengen von Objekten. Damit kann das Konto nur für bestimmte Objekte Berechtigungen verwalten, indem es die ACLs einzelner Objekte aktualisiert.

Wenn Sie ACLs verwenden möchten, um Berechtigungen auf Objektebene zu verwalten, und Sie auch neue Objekte besitzen möchten, die in Ihren Bucket geschrieben wurden, können Sie die bevorzugte Einstellung des Bucket-Eigentümers für Object Ownership anwenden. Ein Bucket mit der bevorzugten Einstellung des Bucket-Eigentümers akzeptiert und berücksichtigt weiterhin Bucket- und Objekt-ACLs. Mit dieser Einstellung gehören neue Objekte, die mit der von `bucket-owner-full-control` vordefinierten ACL geschrieben werden, automatisch dem Bucket-Eigentümer und nicht dem Objekt-Writer. Alle anderen ACL-Verhaltensweisen bleiben bestehen. Damit alle Amazon-S3-PUT-Vorgänge die von `bucket-owner-full-control` vordefinierte ACL enthalten müssen, können Sie eine Bucket-Richtlinie hinzufügen, die nur Objekt-Uploads mit dieser ACL zulässt.

Alternativen zur Verwendung von ACLs

Neben einer Objekt-ACL gibt es noch andere Methoden, wie ein Objekt-Eigentümer Objekt-Berechtigungen verwalten kann:

- Wenn das AWS-Konto , dem das Objekt gehört, auch Eigentümer des Buckets ist, kann es eine Bucket-Richtlinie schreiben, um die Objektberechtigungen zu verwalten.
- Wenn das AWS-Konto , dem das Objekt gehört, einem Benutzer in seinem Konto die Berechtigung erteilen möchte, kann es eine Benutzerrichtlinie verwenden.
- Wenn Sie als Bucket-Eigentümer jedes Objekt in Ihrem Bucket automatisch besitzen und die volle Kontrolle über jedes Objekt in Ihrem Bucket haben möchten, können Sie die erzwungene Einstellung für den Bucket-Eigentümer für Object Ownership anwenden, um ACLs zu deaktivieren.

Infolgedessen basiert die Zugriffskontrolle für Ihre Daten auf Richtlinien. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Verwendung einer Bucket-ACL

Der einzige empfohlene Anwendungsfall für Bucket-ACLs besteht darin, bestimmten AWS-Services wie dem Amazon CloudFront `awslogsdelivery`-Konto Berechtigungen zu erteilen. Wenn Sie eine Verteilung erstellen oder aktualisieren und die CloudFront Protokollierung aktivieren, CloudFront aktualisiert die Bucket-ACL, um dem `awslogsdelivery` Konto `FULL_CONTROL` Berechtigungen zum Schreiben von Protokollen in Ihren Bucket zu erteilen. Weitere Informationen finden Sie unter Erforderliche Berechtigungen für die Konfiguration der Standardprotokollierung und für den Zugriff auf Ihre Protokolldateien im Amazon- CloudFront Entwicklerhandbuch. Wenn der Bucket, der die Protokolle speichert, die Einstellung „Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, um ACLs zu deaktivieren, CloudFront kann keine Protokolle in den Bucket schreiben. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Verwendung einer Bucket-Richtlinie

Wenn ein AWS-Konto , dem ein Bucket gehört, Benutzern in seinem Konto Berechtigungen erteilen möchte, kann er entweder eine Bucket-Richtlinie oder eine Benutzerrichtlinie verwenden. In den folgenden Szenarien müssen Sie eine Bucket-Richtlinie verwenden.

Sie möchten kontoübergreifende Berechtigungen für alle Amazon-S3-Berechtigungen verwalten

Sie können ACLs verwenden, um anderen Konten kontoübergreifende Berechtigungen zu gewähren. ACLs unterstützen jedoch nur einen endlichen Satz von Berechtigungen, die nicht alle Amazon-S3-Berechtigungen enthalten. Weitere Informationen finden Sie unter Welche Berechtigungen kann ich erteilen? Beispielsweise können Sie keine Berechtigungen für Bucket-Subressourcen erteilen. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3.

Sowohl Bucket- als auch Benutzerrichtlinien unterstützen die Erteilung der Berechtigung für alle Amazon-S3-Vorgänge. (Weitere Informationen finden Sie unter Amazon S3-Richtlinienaktionen.) Die Benutzerrichtlinien dienen jedoch der Verwaltung von Berechtigungen für Benutzer in Ihrem Konto. Für kontenübergreifende Berechtigungen für andere AWS-Konten oder Benutzer in einem anderen Konto müssen Sie eine Bucket-Richtlinie verwenden.

Verwendung einer Benutzerrichtlinie

Im Allgemeinen können Sie eine Benutzerrichtlinie oder eine Bucket-Richtlinie verwenden, um Berechtigungen zu verwalten. Sie können Berechtigungen verwalten, indem Sie Benutzer erstellen und Berechtigungen einzeln verwalten, indem Sie Richtlinien an Benutzer (oder Benutzergruppen) anhängen. Oder Sie stellen möglicherweise fest, dass ressourcenbasierte Richtlinien, z. B. eine Bucket-Richtlinie, besser für Ihr Szenario funktionieren.

Mit AWS Identity and Access Management (IAM) können Sie mehrere Benutzer in Ihrem erstellen AWS-Konto und ihre Berechtigungen über Benutzerrichtlinien verwalten. Ein IAM-Benutzer muss über Berechtigungen von dem übergeordneten Konto, zu dem er gehört, und von dem verfügen AWS-Konto , das Eigentümer der Ressource ist, auf die der Benutzer zugreifen möchte. Die Berechtigungen können folgt erteilt werden:

- Berechtigung vom übergeordneten Konto – Das übergeordnete Konto kann seinem Benutzer Berechtigungen erteilen, indem es ihm eine Benutzerrichtlinie zuordnet.
- Berechtigung vom Ressourceneigentümer – Der Ressourceneigentümer kann dem IAM-Benutzer (unter Verwendung einer Bucket-Richtlinie) oder dem übergeordneten Konto (unter Verwendung einer Bucket-Richtlinie, Bucket-ACL oder Objekt-ACL) Berechtigungen erteilen.

Das ist vergleichbar mit einem Kind, das mit einem Spielzeug spielen möchte, das jemand anderem gehört. Das Kind muss die Berechtigung von einem Elternteil erhalten, um mit dem Spielzeug zu spielen, als auch eine Berechtigung vom Eigentümer des Spielzeugs.

Weitere Informationen finden Sie unter Bucket-Richtlinien und Benutzerrichtlinien.

Berechtigungsdelegation

Wenn ein eine Ressource AWS-Konto besitzt, kann er diese Berechtigungen einem anderen erteilen AWS-Konto. Dieses Konto kann diese Berechtigungen oder eine Untermenge davon an Benutzer in dem Konto delegieren. Dies wird auch als Berechtigungsdelegation bezeichnet. Ein Konto, das Berechtigungen von einem anderen Konto erhält, kann die Berechtigungen nicht kontenübergreifend an ein anderes AWS-Konto delegieren.

Verwandte Themen

Wir empfehlen Ihnen, zunächst alle einführenden Themen zu lesen, die erklären, wie Sie den Zugriff auf Ihre Amazon-S3-Ressourcen verwalten, sowie alle zugehörigen Anleitungen. Weitere

Informationen finden Sie unter Identity and Access Management in Amazon S3. Die folgenden Themen enthalten weitere Informationen über spezifische Zugriffsrichtlinienoptionen.

- Zugriffskontrolllisten (ACL) – Übersicht
- Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket.

Wie Amazon S3 eine Anforderung autorisiert

Wenn Amazon S3 eine Anforderung erhält – z. B. ein Bucket oder eine Objektoperation –, überprüft es zuerst, ob der Auftraggeber die erforderlichen Berechtigungen besitzt. Amazon S3 wertet alle relevanten Zugriffsrichtlinien, Benutzerrichtlinien und ressourcenbasierten Richtlinien (Bucket-Richtlinie, Bucket-ACL, Objekt-ACL) aus, um entscheiden zu können, ob die Anforderung autorisiert werden soll.

> **Note**
>
> Wenn bei der Amazon-S3-Berechtigungsprüfung keine gültigen Berechtigungen gefunden werden, wird der Fehler „403-Berechtigung verweigert“ zurückgegeben. Informationen finden Sie unter Beheben Sie Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3.

Um zu ermitteln, ob der Auftraggeber die Berechtigung hat, die spezifische Operation auszuführen, geht Amazon S3 wie folgt vor, wenn es eine Anforderung erhält:

1. Es wandelt alle relevanten Zugriffsrichtlinien (Benutzerrichtlinie, Bucket-Richtlinie, ACLs) zur Laufzeit in eine Richtlinienmenge zur Auswertung um.
2. Es wertet in den folgenden Schritten die resultierende Richtlinienmenge aus. Amazon S3 wertet in jedem Schritt eine Untermenge der Richtlinien in einem spezifischen Kontext aus, basierend auf der Kontextautorität.
   a. Benutzerkontext – Im Benutzerkontext ist das übergeordnete Konto, zu dem der Benutzer gehört, die Kontextautorität.

      Amazon S3 wertet eine Untermenge der Richtlinien aus, die dem übergeordneten Konto gehören. Diese Untermenge beinhaltet die Benutzerrichtlinie, die das übergeordnete Konto dem Benutzer zuordnet. Wenn dem übergeordneten Konto auch die Ressource in der Anforderung gehört (Bucket, Objekt), wertet Amazon S3 gleichzeitig auch die entsprechenden Ressourcenrichtlinien aus (Bucket-Richtlinie, Bucket-ACL und Objekt-ACL).

      Ein Benutzer benötigt die Berechtigung von dem übergeordneten Konto, um die Operation auszuführen.

Dieser Schritt wird nur angewendet, wenn die Anforderung von einem Benutzer in einem AWS-Konto gestellt wurde. Wenn die Anforderung unter Verwendung der Root-Benutzer-Anmeldeinformationen eines erfolgt AWS-Konto, überspringt Amazon S3 diesen Schritt.

b. Bucket-Kontext – Im Bucket-Kontext wertet Amazon S3 Richtlinien aus, die dem gehören AWS-Konto , dem der Bucket gehört.

   Erfolgt die Anforderung für eine Bucket-Operation, muss der Auftraggeber die Berechtigung vom Bucket-Eigentümer besitzen. Erfolgt die Anforderung für ein Objekt, wertet Amazon S3 alle Richtlinien aus, die dem Bucket-Eigentümer gehören, um zu überprüfen, ob der Bucket-Eigentümer für das Objekt eine explizite Zugriffsverweigerung festgelegt hat. Wurde eine explizite Zugriffsverweigerung festgelegt, autorisiert Amazon S3 die Anforderung nicht.

c. Objektkontext – Erfolgt die Anforderung für ein Objekt, wertet Amazon S3 die Untermenge der Richtlinien aus, die dem Objekteigentümer gehören.

Im Folgenden finden Sie einige der Beispielszenarien, die veranschaulichen, wie Amazon S3 eine Anfrage autorisiert.

Example Der Anforderer ist ein IAM-Prinzipal

Wenn der Anforderer ein IAM-Prinzipal ist, muss Amazon S3 bestimmen, ob das übergeordnete , AWS-Konto zu dem der Prinzipal gehört, dem Prinzipal die erforderliche Berechtigung zum Ausführen der Operation erteilt hat. Erfolgt die Anforderung darüber hinaus für eine Bucket-Operation, wie beispielsweise eine Anforderung, den Bucket-Inhalt aufzulisten, muss Amazon S3 prüfen, ob der Bucket-Eigentümer dem Auftraggeber die Berechtigung erteilt hat, die Operation auszuführen.
Um eine bestimmte Operation für eine Ressource auszuführen, benötigt ein IAM-Prinzipal die Berechtigung sowohl von dem übergeordneten , AWS-Konto zu dem er gehört, als auch von dem AWS-Konto , dem die Ressource gehört.

Example Der Anforderer ist ein IAM-Prinzipal – wenn die Anforderung für einen Vorgang an einem Objekt gilt, das der Bucket-Eigentümer nicht besitzt.

Erfolgt die Anforderung für eine Operation für ein Objekt, das nicht dem Bucket-Eigentümer gehört, muss Amazon S3 sicherstellen, dass der Auftraggeber die Berechtigungen von dem Objekteigentümer hat, und außerdem die Bucket-Richtlinie prüfen, um sicherzustellen, dass der Bucket-Eigentümer keine explizite Zugriffsverweigerung für das Objekt festgelegt hat. Ein Bucket-Eigentümer (der die Rechnung zahlt) kann explizit den Zugriff auf Objekte im Bucket verweigern,

unabhängig davon, wem dieser gehört. Der Bucket-Eigentümer kann auch ein beliebiges Objekt im Bucket löschen.

Wenn ein anderes ein Objekt in Ihren S3-Bucket AWS-Konto hochlädt, besitzt dieses Konto (der Objektschreiber) standardmäßig das Objekt, hat Zugriff darauf und kann anderen Benutzern über Zugriffskontrolllisten (ACLs) Zugriff darauf gewähren. Sie können Object Ownership verwenden, um dieses Standardverhalten so zu ändern, dass ACLs deaktiviert sind und Sie als Bucket-Eigentümer automatisch jedes Objekt in Ihrem Bucket besitzen. Daher basiert die Zugriffskontrolle für Ihre Daten auf Richtlinien wie IAM-Benutzerrichtlinien, S3-Bucket-Richtlinien, Endpunktrichtlinien für Virtual Private Cloud (VPC) und AWS Organizations Service-Kontrollrichtlinien (SCPs). Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Weitere Informationen darüber, wie Amazon S3 Zugriffsrichtlinien zur Autorisierung oder Ablehnung von Anfragen für Bucket-Vorgänge und Objekt-Vorgänge bewertet, finden Sie in den folgenden Themen:

Themen

Wie Amazon S3 eine Anforderung für eine Bucket-Operation autorisiert

Wenn Amazon S3 eine Anfrage für eine Bucket-Operation erhält, wandelt Amazon S3 alle relevanten Berechtigungen in eine Reihe von Richtlinien um, die zur Laufzeit ausgewertet werden sollen. Zu den relevanten Berechtigungen gehören ressourcenbasierte Berechtigungen (z. B. Bucket-Richtlinien und Bucket-Zugriffskontrolllisten) und Benutzerrichtlinien, wenn die Anfrage von einem IAM-Prinzipal stammt. Amazon S3 wertet anschließend die resultierende Richtlinienmenge in mehreren Schritten in Übereinstimmung mit einem spezifischen Kontext aus – Benutzerkontext oder Bucket-Kontext.

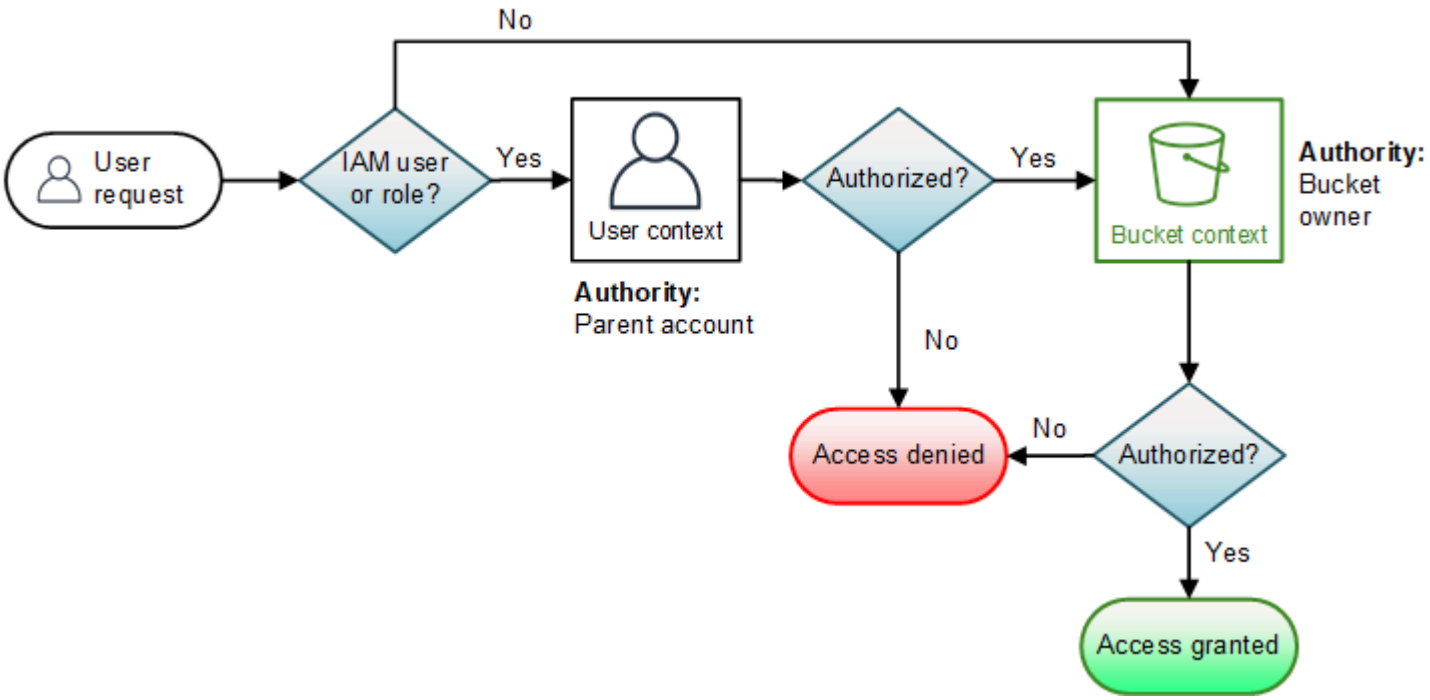
1. Benutzerkontext – Wenn der Anforderer ein IAM-Prinzipal ist, muss der Prinzipal über die Berechtigung von dem übergeordneten verfügen, AWS-Konto zu dem er gehört. In diesem Schritt wertet Amazon S3 eine Untermenge der Richtlinien aus, die dem übergeordneten Konto gehören (auch als Kontextautorität bezeichnet). Diese Richtlinienuntermenge beinhaltet die Benutzerrichtlinie, die das übergeordnete Konto dem Prinzipal zuordnet. Wenn dem übergeordneten Konto auch die Ressource in der Anforderung gehört (in diesem Fall der Bucket), wertet Amazon S3 gleichzeitig auch die entsprechenden Ressourcenrichtlinien aus (Bucket-Richtlinie und Bucket-ACL). Immer wenn eine Anforderung für eine Bucket-

Operation gemacht wird, zeichnen die Server-Zugriffsprotokolle die kanonische Benutzer-ID des Anforderers auf. Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

2. Bucket-Kontext – Der Anforderer muss die Berechtigung vom Bucket-Eigentümer besitzen, eine spezifische Bucket-Operation auszuführen. In diesem Schritt wertet Amazon S3 eine Teilmenge der Richtlinien aus, die dem gehören AWS-Konto , dem der Bucket gehört.

   Der Bucket-Eigentümer kann Berechtigungen unter Verwendung einer Bucket-Richtlinie oder Bucket-ACL erteilen. Beachten Sie, dass das , dem der Bucket gehört AWS-Konto , auch das übergeordnete Konto eines IAM-Prinzipals ist, dann kann es Bucket-Berechtigungen in einer Benutzerrichtlinie konfigurieren.

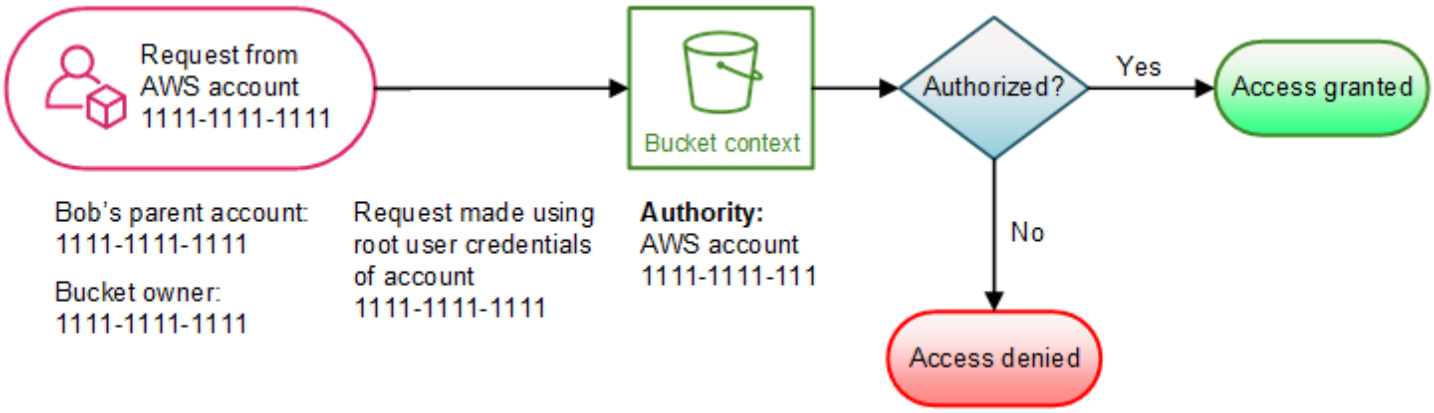
Nachfolgend sehen Sie eine grafische Darstellung der kontextbasierten Auswertung für Bucket-Operationen.



Die folgenden Beispiele veranschaulichen die Auswertungslogik.

Beispiel 1: Vom Bucket-Eigentümer angeforderte Bucket-Operation

In diesem Beispiel sendet der Bucket-Eigentümer eine Anforderung einer Bucket-Operation unter Verwendung der Root-Anmeldeinformationen des AWS-Konto.

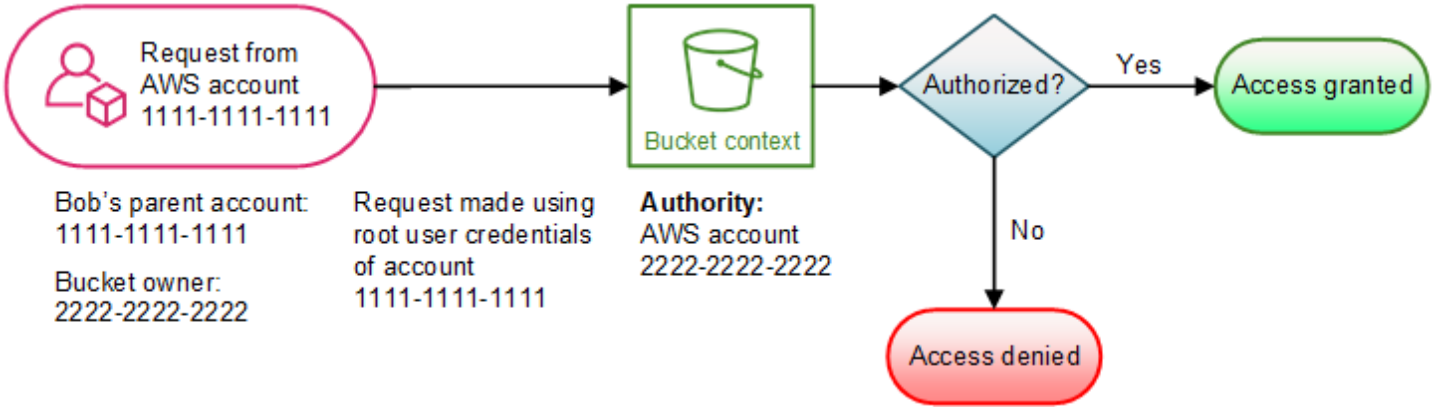


Amazon S3 führt die Kontextauswertung wie folgt durch:

1. Da die Anforderung unter Verwendung der Root-Benutzeranmeldeinformationen eines erfolgt AWS-Konto, wird der Benutzerkontext nicht ausgewertet.
2. Im Bucket-Kontext überprüft Amazon S3 die Bucket-Richtlinie, um festzustellen, ob der Auftraggeber die Berechtigung besitzt, die Operation auszuführen. Amazon S3 autorisiert die Anforderung.

Beispiel 2: Von einem , der nicht der Bucket-Eigentümer ist AWS-Konto , angeforderte Bucket-Operation

In diesem Beispiel wird eine Anforderung unter Verwendung der Root-Benutzeranmeldeinformationen von AWS-Konto 1111-1111-1111 für eine Bucket-Operation gestellt, die im Besitz von AWS-Konto 2222-2222-2222 ist. An dieser Anforderung sind keine IAM-Benutzer beteiligt.



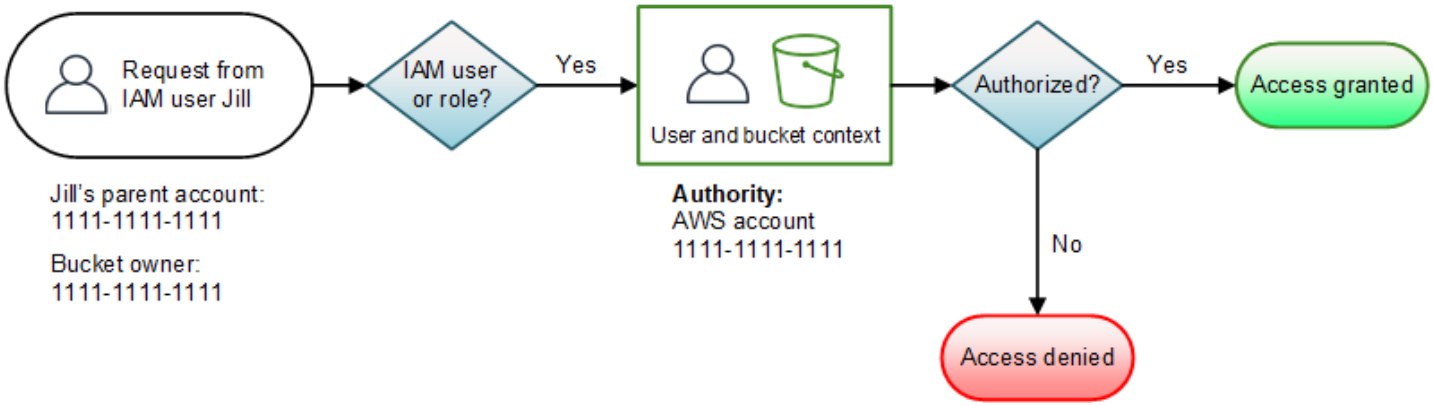
In diesem Fall wertet Amazon S3 den Kontext wie folgt aus:

1. Da die Anforderung unter Verwendung der Root-Benutzeranmeldeinformationen eines erfolgt AWS-Konto, wird der Benutzerkontext nicht ausgewertet.

2. Im Bucket-Kontext wertet Amazon S3 die Bucket-Richtlinie aus. Wenn der Bucket-Eigentümer (AWS-Konto 2222-2222-2222) AWS-Konto 1111-1111-1111 nicht autorisiert hat, den angeforderten Vorgang auszuführen, lehnt Amazon S3 die Anforderung ab. Andernfalls genehmigt Amazon S3 die Anforderung und führt die Operation aus.

Beispiel 3: Von einem IAM-Prinzipal, dessen übergeordnetes Element auch der Bucket-Eigentümer AWS-Konto ist, angeforderte Bucket-Operation

In dem Beispiel wird die Anforderung von Jill gesendet, einer IAM-Benutzer im AWS-Konto 1111-1111-1111, dem auch der Bucket gehört.



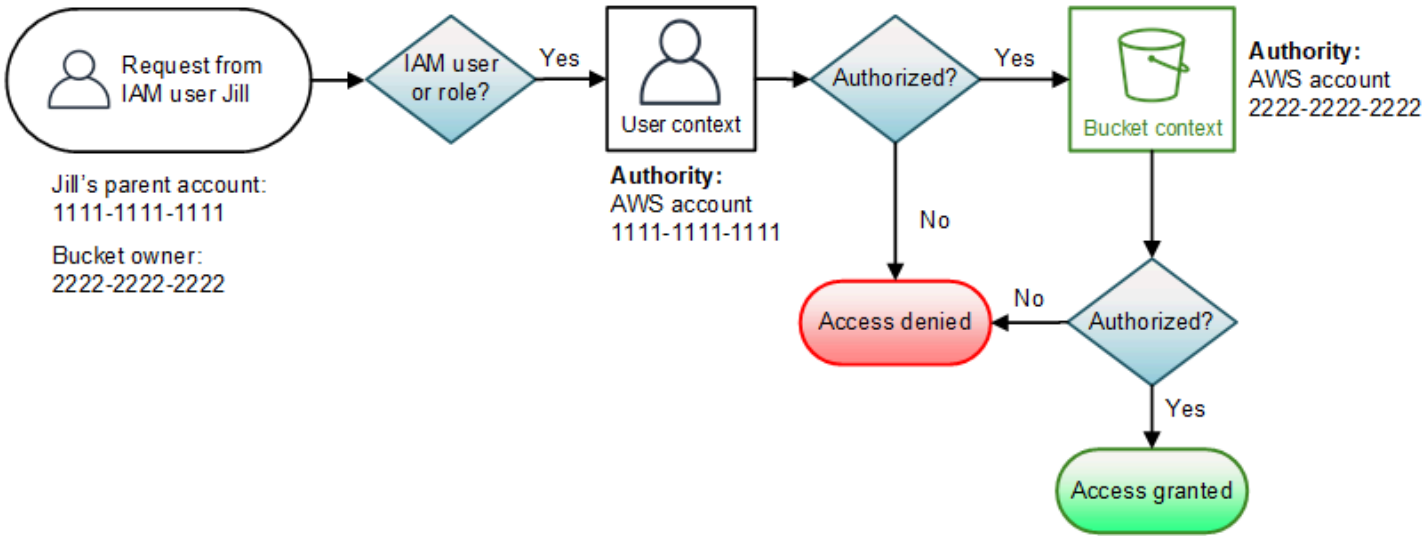
Amazon S3 führt die folgende Kontextauswertung durch:

1. Da die Anfrage von einem IAM-Prinzipal stammt, wertet Amazon S3 im Benutzerkontext alle Richtlinien aus, die zu dem übergeordneten AWS-Konto gehören, um festzustellen, ob Jill zum Ausführen der Operation berechtigt ist.

   In diesem Beispiel ist das übergeordnete AWS-Konto 1111-1111-1111, zu dem der Prinzipal gehört, auch der Bucket-Eigentümer. Somit wertet Amazon S3 zusätzlich zu der Benutzerrichtlinie auch die Bucket-Richtlinie und Bucket-ACL im selben Kontext aus, weil sie zum selben Konto gehören.
2. Amazon S3 hat die Bucket-Richtlinie und die Bucket-ACL als Teil des Benutzerkontexts ausgewertet, deshalb wertet es den Bucket-Kontext nicht aus.

Beispiel 4: Von einem IAM-Prinzipal, dessen übergeordnetes nicht der Bucket-Eigentümer AWS-Konto ist, angeforderte Bucket-Operation

In diesem Beispiel wird die Anforderung von Jill gesendet, einer IAM-Benutzerin, deren übergeordnetes Element 1111-1111-1111 AWS-Konto ist, aber der Bucket gehört einem anderen AWS-Konto, 2222-2222-2222.



Jill benötigt Berechtigungen sowohl vom übergeordneten als auch AWS-Konto vom Bucket-Eigentümer. Amazon S3 wertet den Kontext wie folgt aus:

1. Da die Anforderung von einem IAM-Prinzipal stammt, wertet Amazon S3 den Benutzerkontext aus, indem es die Richtlinien überprüft, die vom Konto autorisiert wurden, um sicherzustellen, dass Jill über die erforderlichen Berechtigungen verfügt. Wenn Jill die Berechtigung besitzt, wertet Amazon S3 den Bucket-Kontext aus, andernfalls weist es die Anforderung ab.
2. Im Bucket-Kontext überprüft Amazon S3, ob der Bucket-Eigentümer 2222-2222-2222 Jill (oder ihrem übergeordneten AWS-Konto) die Berechtigung erteilt hat, den angeforderten Vorgang auszuführen. Wenn sie diese Berechtigung besitzt, genehmigt Amazon S3 die Anforderung und führt die Operation aus. Andernfalls lehnt Amazon S3 die Anforderung ab.

Wie Amazon S3 eine Anforderung für eine Objekt-Operation autorisiert

Wenn Amazon S3 eine Anforderung für eine Bucket-Operation erhält, wandelt es alle relevanten Berechtigungen – ressourcenbasierte Berechtigungen (Objekt-Access-Control-List (ACL), Bucket-Richtlinie, Bucket-ACL) und IAM-Benutzerrichtlinien – in eine Reihe von Richtlinien um, die zur Laufzeit ausgewertet werden sollen. Anschließend wertet es in mehreren Schritten die resultierende

Richtlinienmenge aus. Es wertet in jedem Schritt eine Untermenge der Richtlinien in drei spezifischen Kontexten aus – Benutzerkontext, Bucket-Kontext und Objektkontext.

1. Benutzerkontext – Wenn der Anforderer ein IAM-Prinzipal ist, muss der Prinzipal über die Berechtigung von dem übergeordneten verfügen, AWS-Konto zu dem er gehört. In diesem Schritt wertet Amazon S3 eine Untermenge der Richtlinien aus, die dem übergeordneten Konto gehören (auch als Kontextautorität bezeichnet). Diese Richtlinienuntermenge beinhaltet die Benutzerrichtlinie, die das übergeordnete Konto dem Prinzipal zuordnet. Wenn dem übergeordneten Konto auch die Ressource in der Anforderung gehört (Bucket, Objekt), wertet Amazon S3 gleichzeitig die entsprechenden Ressourcenrichtlinien aus (Bucket-Richtlinie, Bucket-ACL und Objekt-ACL).

   > **Note**
   >
   > Wenn das übergeordnete AWS-Konto Eigentümer der Ressource (Bucket oder Objekt) ist, kann es seinem IAM-Prinzipal mithilfe der Benutzerrichtlinie oder der Ressourcenrichtlinie Ressourcenberechtigungen erteilen.

2. Bucket-Kontext – In diesem Kontext wertet Amazon S3 Richtlinien aus, die dem AWS-Konto gehören, dem der Bucket gehört.

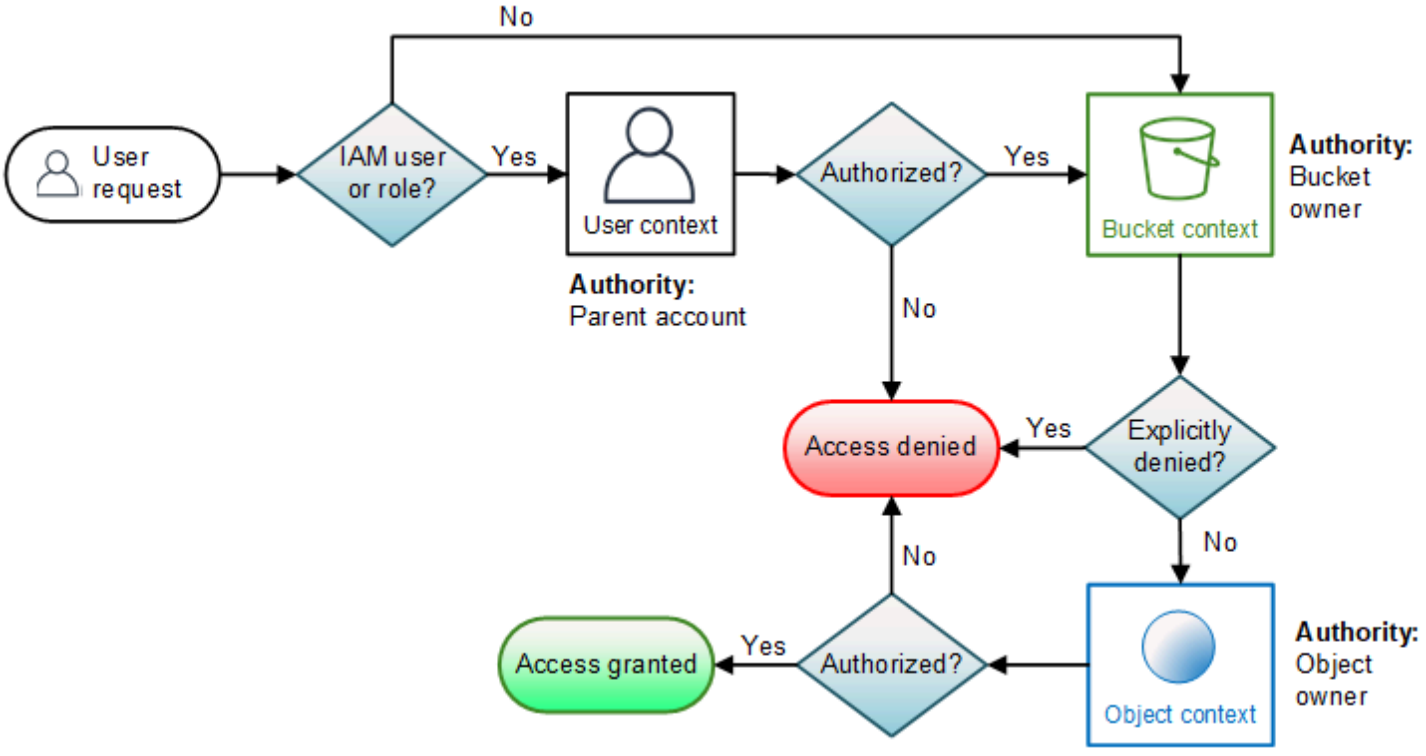
   Wenn das AWS-Konto , dem das Objekt in der Anforderung gehört, nicht mit dem Bucket-Eigentümer übereinstimmt, überprüft Amazon S3 im Bucket-Kontext die Richtlinien, ob der Bucket-Eigentümer den Zugriff auf das Objekt explizit verweigert hat. Wurde eine explizite Zugriffsverweigerung für das Objekt festgelegt, autorisiert Amazon S3 die Anforderung nicht.

3. Objekt-Kontext – Der Anforderer muss die Berechtigung vom Objekt-Eigentümer besitzen, eine spezifische Objekt-Operation auszuführen. In diesem Schritt wertet Amazon S3 die Objekt-ACL aus.

   > **Note**
   >
   > Ist der Bucket-Eigentümer gleich dem Objekt-Eigentümer ist, kann der Zugriff auf das Objekt in der Bucket-Richtlinie erteilt werden, die im Bucket-Kontext ausgewertet wird. Unterscheiden sich die Eigentümer, müssen die Objekt-Eigentümer eine Objekt-ACL verwenden, um die Berechtigungen zu erteilen. Wenn das AWS-Konto , dem das Objekt gehört, auch das übergeordnete Konto ist, zu dem der IAM-Prinzipal gehört, kann er Objektberechtigungen in einer Benutzerrichtlinie konfigurieren, die im Benutzerkontext

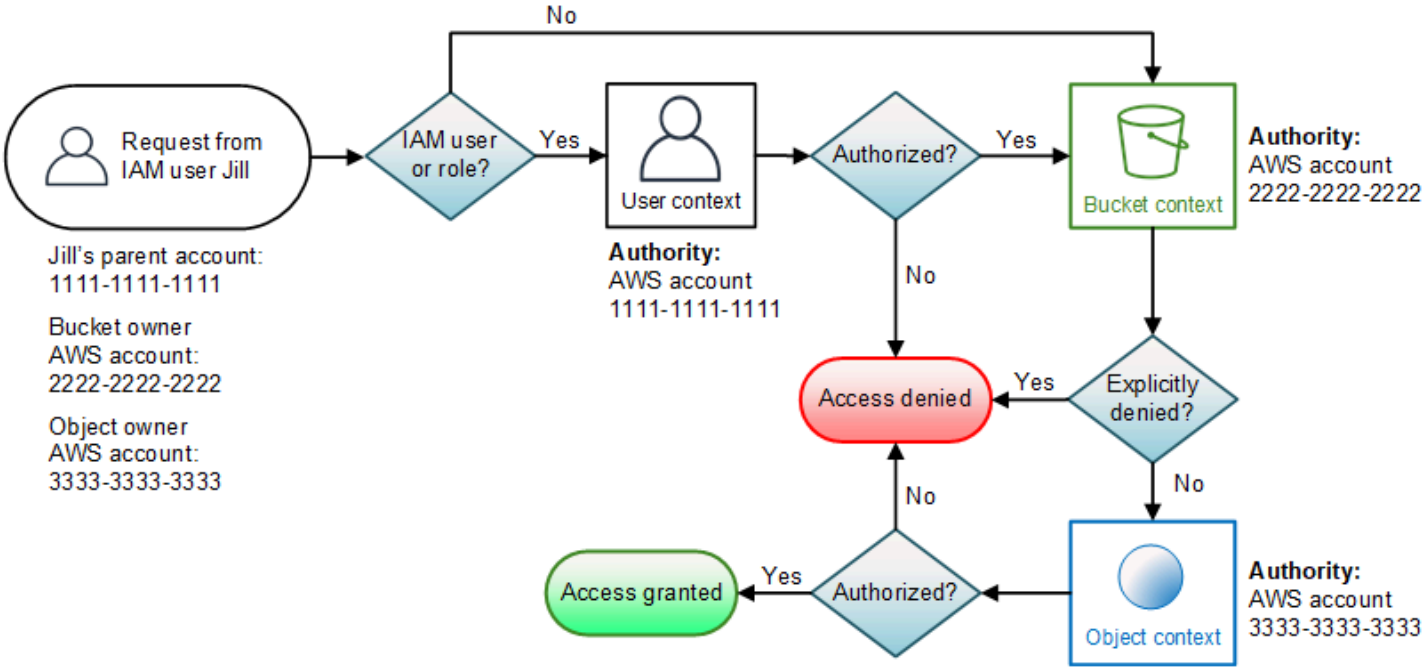
ausgewertet wird. Weitere Informationen über die Verwendung dieser Alternativen zu Zugriffsrichtlinien finden Sie unter Richtlinien für Zugriffsrichtlinien.
Wenn Sie als Bucket-Eigentümer alle Objekte in Ihrem Bucket besitzen und Bucket-Richtlinien oder IAM-basierte Richtlinien verwenden möchten, um den Zugriff auf diese Objekte zu verwalten, können Sie die erzwungene Einstellung des Bucket-Eigentümers für Object Ownership anwenden. Mit dieser Einstellung besitzen Sie als Bucket-Eigentümer automatisch die volle Kontrolle über jedes Objekt in Ihrem Bucket. Bucket- und Objekt-ACLs können nicht bearbeitet werden und werden nicht mehr für den Zugriff berücksichtigt. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Nachfolgend sehen Sie eine grafische Darstellung der kontextbasierten Auswertung für eine Objekt-Operation.



Beispiel 1: Anforderung einer Objekt-Operation

In diesem Beispiel sendet die IAM-Benutzerin Jill, deren übergeordnetes Element 1111-1111-1111 AWS-Konto ist, eine Anforderung für eine Objektoperation (z. B. Get object) für ein Objekt, das AWS-Konto 3333-3333-3333 gehört, in einem Bucket, das AWS-Konto 2222-2222-2222 gehört.



Jill benötigt die Berechtigung von der übergeordneten AWS-Konto, dem Bucket-Eigentümer und dem Objekteigentümer. Amazon S3 wertet den Kontext wie folgt aus:

1. Da die Anforderung von einem IAM-Prinzipal stammt, wertet Amazon S3 den Benutzerkontext aus, um zu überprüfen, ob das übergeordnete AWS-Konto 1111-1111-1111 Jill die Berechtigung zum Ausführen der angeforderten Operation erteilt hat. Wenn sie die Berechtigung besitzt, wertet Amazon S3 den Bucket-Kontext aus. Andernfalls lehnt Amazon S3 die Anforderung ab.
2. Im Bucket-Kontext ist der Bucket-Eigentümer, das AWS-Konto 2222-2222-2222, die Kontextautorität. Amazon S3 wertet die Bucket-Richtlinie aus, um festzustellen, ob der Bucket-Eigentümer Jill explizit die Berechtigung entzogen hat, auf das Objekt zuzugreifen.
3. Im Objektkontext ist die Kontextautorität das AWS-Konto 3333-3333-3333, der Objekteigentümer. Amazon S3 wertet die Objekt-ACL aus, um festzustellen, ob Jill die Berechtigung besitzt, auf das Objekt zuzugreifen. Ist dies der Fall, autorisiert Amazon S3 die Anfrage.

Bucket-Richtlinien und Benutzerrichtlinien

Bucket-Richtlinien und Benutzerrichtlinien sind zwei Zugriffsrichtlinienoptionen für die Erteilung von Berechtigungen für Ihre Amazon-S3-Ressourcen. Beide verwenden die JSON-basierte Zugriffsrichtliniensprache.

Die Themen in diesem Abschnitt beschreiben die Schlüsselelemente der Richtliniensprache mit Schwerpunkt auf für Amazon S3 spezifischen Details und bieten Beispiele für Bucket- und Benutzerrichtlinien. Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die Grundkonzepte und die für Sie verfügbaren Optionen zum Verwalten des Zugriffs auf Ihre Amazon-S3-Ressourcen erläutert werden. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3.

> **Important**
>
> Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt.

Themen

Richtlinien und Berechtigungen in Amazon S3

Diese Seite bietet eine Übersicht über Bucket- und Benutzerrichtlinien in Amazon S3 und beschreibt die Basiselemente einer Richtlinie. Jedes aufgelistete Element verweist auf weitere Details zu diesem Element und auf Beispiele für die Verwendung dieses Elements.

Eine vollständige Liste der Amazon S3-Aktionen, -Ressourcen und -Bedingungen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

In ihrer einfachsten Form enthält eine Richtlinie die folgenden Elemente:

- Ressource: Der Amazon-S3-Bucket, das Objekt, der Zugangspunkt oder der Auftrag, für den die Richtlinie gilt. Verwenden Sie den Amazon-Ressourcennamen (ARN) des Buckets, Objekts, Zugangspunkts oder Auftrags, um die Ressource zu identifizieren.

  Ein Beispiel für Operationen auf Bucket-Ebene:

  - `"Resource": "arn:aws:s3:::bucket_name"`.

  Beispiele für Operationen auf Objektebene:

  - `"Resource": "arn:aws:s3:::bucket_name/*"` aller Objekte im Bucket.

  - `"Resource": "arn:aws:s3:::bucket_name/prefix/*"` für Objekte unter einem bestimmten Präfix im Bucket.

  Weitere Informationen finden Sie unter Amazon-S3-Ressourcen.

- Aktionen – Für jede Ressource unterstützt Amazon S3 eine Reihe von Vorgänge. Sie identifizieren RessourcenVorgänge, die Sie zulassen (oder ablehnen) können, indem Sie Aktionsschlüsselwörter verwenden.

  Beispielsweise ermöglicht die Berechtigung `s3:ListBucket` dem Benutzer die Verwendung der Amazon-S3-Operation GET Bucket (List Objects). Weitere Informationen zur Verwendung von Amazon-S3-Aktionen finden Sie unter Amazon S3-Richtlinienaktionen. Eine vollständige Liste der Amazon-S3-Aktionen finden Sie unter Aktionen.

- Auswirkung – zeigt die Auswirkung, wenn ein Benutzer die spezifische Aktion anfordert – entweder Allow (Zugriffserlaubnis) oder Deny (Zugriffsverweigerung).

  Wenn Sie den Zugriff auf eine Ressource nicht ausdrücklich gestatten ("allow"), ist der Zugriff automatisch verweigert. Sie können den Zugriff auf eine Ressource auch explizit verweigern. So könnten Sie z. B. sicherstellen, dass ein Benutzer nicht auf die Ressource zugreifen kann, auch wenn der Zugriff durch eine andere Richtlinie gestattet wird. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Auswirkung.

- Prinzipal – Das Konto oder der Benutzer, das oder der Zugriff auf die Aktionen und Ressourcen in der Anweisung hat. In einer Bucket-Richtlinie ist der Prinzipal der Benutzer, das Konto, der Service oder eine andere Entität, der/die/das der Empfänger dieser Berechtigung ist. Weitere Informationen finden Sie unter Prinzipale.

- Bedingung – Bedingungen für den Zeitpunkt, an dem eine Richtlinie in Kraft ist. Sie können AWS-weite Schlüssel und Amazon S3-spezifische Schlüssel verwenden, um Bedingungen in einer

Amazon S3-Zugriffsrichtlinie anzugeben. Weitere Informationen finden Sie unter Beispiele für Amazon-S3-Bedingungsschlüssel.

Die folgende beispielhafte Bucket-Richtlinie zeigt die Elemente Auswirkung, Prinzipal, Aktion und Ressource. Die Richtlinie gewährt Akua, einem Benutzer in *Konto-ID* , `s3:GetObjects3:GetBucketLocation`, und `s3:ListBucket` Amazon S3 Berechtigungen für den `awsexamplebucket1` Bucket.

```
{
    "Version": "2012-10-17",
    "Id": "ExamplePolicy01",
    "Statement": [
        {
            "Sid": "ExampleStatement01",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Akua"
            },
            "Action": [
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::awsexamplebucket1/*",
                "arn:aws:s3:::awsexamplebucket1"
            ]
        }
    ]
}
```

Weitere Informationen finden Sie in den folgenden Themen. Ausführliche Informationen zur Richtliniensprache finden Sie unter Richtlinien und Berechtigungen und IAM-JSON-Richtlinienreferenz im IAM-Benutzerhandbuch.

Themen

- Amazon-S3-Ressourcen
- Prinzipale
- Amazon S3-Richtlinienaktionen

- Amazon-S3-Bedingungsschlüssel

Amazon-S3-Ressourcen

Das folgende allgemeine Format für Amazon-Ressourcennamen (ARNs) identifiziert Ressourcen in AWS:

```
arn:partition:service:region:namespace:relative-id
```

Weitere Informationen zu ARNs finden Sie unter Amazon-Ressourcennamen (ARNs) in der Allgemeine AWS-Referenz.

Weitere Informationen zu Ressourcen finden Sie unter IAM-JSON-Richtlinienelemente: Ressource im IAM-Benutzerhandbuch.

Ein Amazon S3-ARN schließt den - AWS-Region und -Namespace aus, enthält jedoch Folgendes:

- Partition – `aws` ist ein allgemeiner Partitionsname. Wenn sich Ihre Ressourcen in der Region China (Peking) befinden, ist `aws-cn` der Partitionsname.
- Service - `s3`.
- Relative ID - `bucket-name` oder eine `bucket-name/object-key`. Sie können Platzhalter verwenden.

Das ARN-Format für Amazon-S3-Ressourcen reduziert sich auf:

```
arn:aws:s3:::bucket_name/key_name
```

Eine vollständige Liste der Amazon S3-Ressourcen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Um den ARN für einen S3-Bucket zu finden, können Sie sich die Berechtigungsseiten Bucket Policy (Bucket-Richtlinie) oder CORS configuration (CORS-Konfiguration) in der Amazon-S3-Konsole ansehen. Weitere Informationen finden Sie unter den folgenden Themen:

- Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole
- CORS-Konfiguration

Amazon S3 ARN-Beispiele

Es folgen Beispiele für Amazon-S3-Ressourcen-ARNs.

Bucket-Name und Objektschlüssel angegeben

Der folgende ARN gibt das Objekt `/developers/design_info.doc` im Bucket `examplebucket` an.

```
arn:aws:s3:::examplebucket/developers/design_info.doc
```

Platzhalter

Sie können im ARN der Ressource Platzhalter verwenden. Sie können Platzhalterzeichen (`*` und `?`) in einem beliebigen ARN-Segment (durch Doppelpunkte getrennte Teile) verwenden. Ein Sternchen (`*`) steht für kein Zeichen oder eine beliebige Kombination von mehreren Zeichen und ein Fragezeichen (`?`) entspricht einem beliebigen einzelnen Zeichen. Sie können mehrere `*`- oder `?`-Zeichen in jedem Segment verwenden, allerdings sind Platzhalter nicht segmentübergreifend.

- Der folgende ARN verwendet den Platzhalter `*` im Teil mit der relativen ID des ARN, um alle Objekte im Bucket `examplebucket` anzugeben.

  ```
  arn:aws:s3:::examplebucket/*
  ```

- Der folgende ARN verwendet `*`, um alle Amazon-S3-Ressourcen (alle S3-Buckets und Objekte) anzugeben.

  ```
  arn:aws:s3:::*
  ```

- Der folgende ARN verwendet beide Platzhalter, `*` und `?`, im Teil `relative-ID`. Er identifiziert alle Objekte in Buckets wie `example1bucket`, `example2bucket`, `example3bucket` usw.

  ```
  arn:aws:s3:::example?bucket/*
  ```

Richtlinienvariablen

Sie können Richtlinienvariablen in Amazon-S3-ARNs verwenden. Bei der Richtlinienauswertung werden diese vordefinierten Variablen durch ihre entsprechenden Werte ersetzt. Angenommen, Sie organisieren Ihren Bucket als eine Sammlung von Ordnern, je ein Ordner für jeden Ihrer Benutzer.

Der Ordnername entspricht dabei dem Benutzernamen. Um den Benutzern Berechtigungen für ihre Ordner zu erteilen, können Sie eine Richtlinienvariable im Ressourcen-ARN angeben:

```
arn:aws:s3:::bucket_name/developers/${aws:username}/
```

Zur Laufzeit wird bei der Auswertung der Richtlinie die Variable `${aws:username}` im Ressourcen-ARN durch den Benutzernamen ersetzt, der die Anforderung stellt.

Prinzipale

Das Element `Principal` gibt an, welchem Benutzer, Konto, Dienst oder welcher anderen Entität der Zugriff auf eine Ressource gewährt oder verweigert wird. Die folgenden sind Beispiele legen den `Principal` fest. Weitere Informationen finden Sie unter Prinzipal im IAM-Benutzerhandbuch.

Erteilen von Berechtigungen für ein AWS-Konto

Um einem Berechtigungen zu erteilen AWS-Konto, identifizieren Sie das Konto im folgenden Format.

```
"AWS":"account-ARN"
```

Im Folgenden sind einige Beispiele aufgeführt.

```
"Principal":{"AWS":"arn:aws:iam::AccountIDWithoutHyphens:root"}
```

```
"Principal":{"AWS":
["arn:aws:iam::AccountID1WithoutHyphens:root","arn:aws:iam::AccountID2WithoutHyphens:root"]}
```

Amazon S3 unterstützt auch eine kanonische Benutzer-ID, eine verschleierte Form der AWS-Konto ID. Sie können diese ID im folgenden Format angeben.

```
"CanonicalUser":"64-digit-alphanumeric-value"
```

Im Folgenden wird ein Beispiel gezeigt.

```
"Principal":{"CanonicalUser":"64-digit-alphanumeric-value"}
```

Informationen darüber, wo Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Suchen Ihrer kanonischen Kontobenutzer-ID.

> **Important**
>
> Wenn Sie eine kanonische Benutzer-ID in einer Richtlinie verwenden, ändert Amazon S3 möglicherweise die kanonische ID in die entsprechende AWS-Konto ID. Dies hat keine Auswirkungen auf die Richtlinie, da beide IDs dasselbe Konto identifizieren.

Erteilen von Berechtigungen für einen IAM-Benutzer

Um einem IAM-Benutzer in Ihrem Konto eine Berechtigung zu erteilen, müssen Sie ein `"AWS":"`*`user-ARN`*`"`-Name-Wert-Paar bereitstellen.

```
"Principal":{"AWS":"arn:aws:iam::account-number-without-hyphens:user/username"}
```

Ausführliche Beispiele, die step-by-step Anweisungen enthalten, finden Sie unter Beispiel 1: Bucket-Eigentümer erteilt seinen Benutzern Bucket-Berechtigungen und Beispiel 3: Bucket-Eigentümer, der Berechtigungen für Objekte erteilt, die ihm nicht gehören.

> **Note**
>
> Wenn eine IAM-Identität gelöscht wird, nachdem Sie Ihre Bucket-Richtlinie aktualisiert haben, zeigt die Bucket-Richtlinie anstelle eines ARN eine eindeutige Kennung im Hauptelement an. Diese eindeutigen IDs werden niemals wiederverwendet, sodass Sie Prinzipale mit eindeutigen Identifikatoren ohne Risiko aus all Ihren Versicherungserklärungen entfernen können. Weitere Informationen zu eindeutigen Kennungen finden Sie unter IAM-Kennungen im IAM-Benutzerhandbuch.

Erteilen anonymer Berechtigungen

Um jedem Benutzer Berechtigung zu erteilen, auch als anonymer Zugriff bezeichnet, legen Sie den Platzhalter (`"*"`) auf den Wert `Principal` fest. Wenn Sie beispielsweise ihren Bucket als Website konfigurieren, müssen Sie alle Objekte im Bucket öffentlich zugänglich machen.

```
"Principal":"*"
```

```
"Principal":{"AWS":"*"}
```

Die Verwendung von `"Principal": "*"` mit einem -`Allow`Effekt in einer ressourcenbasierten Richtlinie ermöglicht es jedem, auch wenn er nicht bei angemeldet ist AWS, auf Ihre Ressource zuzugreifen.

Das Verwenden von `"Principal" : { "AWS" : "*" }` mit einem `Allow`-Effekt in einer ressourcenbasierten Richtlinie ermöglicht jeden Stammbenutzer, IAM-Benutzer, angenommener Rollensitzung oder Verbundbenutzer in einem beliebigen Konto in der selben Partition, auf Ihre Ressource zuzugreifen.

Für anonyme Benutzer sind diese beiden Methoden gleichwertig. Weitere Informationen finden Sie unter Alle Prinzipale im IAM-Benutzerhandbuch.

Sie können keinen Platzhalter verwenden, um einen Teil eines Namens oder eines ARNs zu ersetzen.

> **Important**
>
> Da jeder ein erstellen kann AWS-Konto, entspricht die Sicherheitsstufe dieser beiden Methoden, obwohl sie unterschiedlich funktionieren.

> **Warning**
>
> Seien Sie vorsichtig, wenn Sie anonymen Zugriff auf Ihren Amazon-S3-Bucket erteilen. Wenn Sie anonymen Zugriff gewähren, kann jeder auf der ganzen Welt auf Ihren Bucket zugreifen. Wir empfehlen dringend, nie einen anonymen Schreibzugriff auf Ihren S3-Bucket zu gewähren.

Ressourcenberechtigungen beschränken

Sie können die Ressourcenrichtlinie auch verwenden, um den Zugriff auf Ressourcen einzuschränken, die sonst für IAM-Prinzipale verfügbar wären. Verwenden Sie eine `Deny`-Anweisung, um den Zugriff zu verhindern.

Das folgende Beispiel blockiert den Zugriff, wenn kein sicheres Transportprotokoll verwendet wird:

```
{"Effect": "Deny",
 "Principal": "*",
 "Action": "s3:*",
```

```
 "Resource": <bucket ARN>,
 "Condition": {
   "Boolean": { "aws:SecureTransport" : "false"}
 }
}
```

Es hat sich für diese Richtlinie bewährt, `"Principal": "*"` so zu verwenden, dass diese Einschränkung für alle gilt, anstatt zu versuchen, mit dieser Methode nur bestimmten Konten oder Prinzipalen den Zugriff zu verweigern.

Erzwingen des Zugriffs über CloudFront URLs

Sie können verlangen, dass Ihre Benutzer auf Ihre Amazon S3-Inhalte zugreifen, indem Sie Amazon CloudFront-URLs anstelle von Amazon S3-URLs verwenden. Erstellen Sie dazu eine CloudFront Ursprungszugriffsidentität (OAI). Anschließend ändern Sie die Berechtigungen für Ihren Bucket oder für die Objekte in Ihrem Bucket. Das Format für die Angabe des OAI in einer `Principal`-Anweisung ist wie folgt.

```
"Principal":{"CanonicalUser":"Amazon S3 Canonical User ID assigned to origin access
 identity"}
```

Weitere Informationen finden Sie unter Verwenden einer Ursprungszugriffsidentität zum Einschränken des Zugriffs auf Ihre Amazon S3-Inhalte im Amazon- CloudFront Entwicklerhandbuch.

Amazon S3-Richtlinienaktionen

> **Note**
>
> Auf dieser Seite geht es um Amazon S3-Richtlinienaktionen für Allzweck-Buckets. Weitere Informationen zu Amazon S3-Richtlinienaktionen für Verzeichnis-Buckets finden Sie unter Aktionen für S3 Express One Zone.

Amazon S3 definiert eine Reihe von Berechtigungen, die Sie in einer Richtlinie angeben können. Um Berechtigungen zum Ausführen einer S3-API-Operation zu erteilen, müssen Sie eine gültige Richtlinie (z. B. eine S3-Bucket-Richtlinie oder eine identitätsbasierte IAM-Richtlinie) erstellen und entsprechende Aktionen im -`Action`Element der Richtlinie angeben. Diese Aktionen werden als Richtlinienaktionen bezeichnet. Im Folgenden werden verschiedene Arten von Zuordnungsbeziehungen zwischen S3-API-Operationen und den erforderlichen Richtlinienaktionen gezeigt.

- One-to-one -Zuweisung mit demselben Namen. Um beispielsweise die `PutBucketPolicy`-API-Operation zu verwenden, ist die `s3:PutBucketPolicy` Richtlinienaktion erforderlich.
- One-to-one -Zuweisung mit unterschiedlichen Namen. Um beispielsweise die `ListObjectsV2`-API-Operation zu verwenden, ist die `s3:ListBucket` Richtlinienaktion erforderlich.
- O-ne-to-many Zuweisung. Um beispielsweise die `HeadObject`-API-Operation zu verwenden, `s3:GetObject` ist erforderlich. Wenn Sie die S3-Objektsperre verwenden und den Status oder die Aufbewahrungseinstellungen für rechtliche Aufbewahrungsfristen eines Objekts abrufen möchten, sind auch die entsprechenden - `s3:GetObjectLegalHold` oder -`s3:GetObjectRetention`Richtlinienaktionen erforderlich, bevor Sie die `HeadObject` API-Operation verwenden können.
- M-any-to-one Zuweisung. Um beispielsweise die - `ListObjectsV2` oder `HeadBucket`-API-Operationen zu verwenden, ist die `s3:ListBucket` Richtlinienaktion erforderlich.

Um eine gültige S3-Bucket-Richtlinie zu erstellen, müssen Sie neben dem -`Action`Element auch die `Resource` Elemente `EffectPrincipal`, und angeben. Darüber hinaus können Sie das -`Condition`Element angeben, um eine detailliertere Kontrolle über S3-API-Operationen zu haben.

Um eine gültige identitätsbasierte IAM-Richtlinie zu erstellen, müssen Sie neben dem -`Action`Element auch - `Effect` und -`Resource`Elemente angeben. Eine gültige identitätsbasierte IAM-Richtlinie enthält das -`Principal`Element nicht.

Eine vollständige Liste der Amazon S3-Richtlinienaktionen, Ressourcen und Bedingungsschlüssel für die Verwendung in -Richtlinien finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Wenn Sie Richtlinien erstellen, müssen Sie das -`Resource`Element auf der Grundlage des richtigen Ressourcentyps angeben, der für die entsprechenden Amazon S3-Richtlinienaktionen erforderlich ist. Auf dieser Seite werden Berechtigungen für S3-API-Operationen nach den Ressourcentypen kategorisiert. Weitere Informationen zu den Ressourcentypen finden Sie unter Von Amazon S3 definierte Ressourcentypen in der Service-Autorisierungs-Referenz. Eine vollständige Liste der Amazon S3-API-Operationen finden Sie unter Amazon S3-API-Aktionen in der Amazon-Simple-Storage-Service-API-Referenz.

Themen

- Zugriffspunkt-Operationen
- Operationen für Object Lambda Access Points
- Operationen für Multi-Region Access Points
- Batch-Auftragsoperationen
- S3-Storage-Lens-Konfigurationsvorgänge
- Kontooperationen

Bucket-Operationen

Bucket-Operationen sind S3-API-Operationen, die für den Bucket-Ressourcentyp ausgeführt werden. Beispiel: `CreateBucket`, `ListObjectsV2` und `PutBucketPolicy`. S3-Richtlinienaktionen für Bucket-Operationen erfordern, dass das -`Resource`Element in Bucket-Richtlinien oder identitätsbasierten IAM-Richtlinien die Amazon-Ressourcenname (ARN)-ID des S3-Bucket-Typs im folgenden Beispielformat ist.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
```

Die folgende Bucket-Richtlinie erteilt dem Benutzer *Akua* mit Konto *12345678901* die `s3:ListBucket` Berechtigung, die ListObjectsV2-API-Operation auszuführen und Objekte in einem S3-Bucket aufzulisten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to list objects in the bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        }
    ]
}
```

Bucket-Operationen in Zugriffspunktrichtlinien

Berechtigungen, die in einer Zugriffspunktrichtlinie gewährt werden, sind nur wirksam, wenn der zugrunde liegende Bucket dieselben Berechtigungen zulässt. Wenn Sie S3 Access Points verwenden, müssen Sie die Zugriffskontrolle vom Bucket an den Zugriffspunkt delegieren oder der Richtlinie des zugrunde liegenden Buckets dieselben Berechtigungen in den Zugriffspunktrichtlinien hinzufügen. Weitere Informationen finden Sie unter Konfigurieren von IAM-Richtlinien für die Verwendung von Zugriffspunkten. In Zugriffspunktrichtlinien erfordern S3-Richtlinienaktionen für Bucket-Operationen die Verwendung des `accesspoint` ARN für das -`Resource`Element im folgenden Format.

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT"
```

Die folgende Zugriffspunktrichtlinie gewährt dem Benutzer *Akua* mit Konto *12345678901* die `s3:ListBucket` Berechtigung, die ListObjectsV2-API-Operation über den S3-Zugriffspunkt *DOC-EXAMPLE-ACCESS-POINT* auszuführen, um Objekte im zugehörigen Bucket des Zugriffspunkts aufzulisten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to list objects in the bucket through access point",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-
ACCESS-POINT"
        }
    ]
}
```

> **Note**
>
> Nicht alle Bucket-Operationen werden von S3 Access Point unterstützt. Weitere Informationen finden Sie unter Zugriffspunkt-Kompatibilität mit S3-Vorgänge.

Objektoperationen

Objektoperationen sind S3-API-Operationen, die auf den Objektressourcentyp wirken. Beispiel: `GetObject`, `PutObject` und `DeleteObject`. S3-Richtlinienaktionen für Objektoperationen erfordern, dass das -`Resource`Element in Richtlinien in den folgenden Beispielformaten der S3-Objekt-ARN ist.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
```

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/*"
```

> **Note**
>
> Der Objekt-ARN muss einen Schrägstrich nach dem Bucket-Namen enthalten, wie in den vorherigen Beispielen gezeigt.

Die folgende Bucket-Richtlinie gewährt dem Benutzer *Akua* mit Konto *12345678901* die `s3:PutObject` Berechtigung, den PutObject API-Vorgang zum Hochladen von Objekten in einen S3-Bucket auszuführen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to upload objects",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Objektoperationen in Zugriffspunktrichtlinien

Wenn Sie den Zugriff auf Objektoperationen mit S3 Access Points steuern, können Sie Zugriffspunktrichtlinien verwenden. Wenn Sie Zugriffspunktrichtlinien verwenden, erfordern S3-Richtlinienaktionen für Objektoperationen die Verwendung des `accesspoint` ARN für das -`Resource`Element im folgenden Format: `arn:aws:s3:region:account-id:accesspoint/access-point-name/object/resource`. Für Objektoperationen, die Zugriffspunkt verwenden, müssen Sie den `/object/` Wert nach dem gesamten Zugriffspunkt-ARN in das -`Resource`Element aufnehmen. Hier sind einige Beispiele.

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT/object/*"
```

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT/object/prefix/*"
```

Die folgende Zugriffspunktrichtlinie gewährt dem Benutzer *`Akua`* mit Konto *`12345678901`* die `s3:GetObject` Berechtigung, die GetObject API-Operation über den Zugriffspunkt *`DOC-EXAMPLE-ACCESS-POINT`* für alle Objekte im zugehörigen Bucket des Zugriffspunkts auszuführen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to get objects through access point",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT/object/*"
        }
    ]
}
```

> **Note**
>
> Nicht alle Objektoperationen werden von S3 Access Point unterstützt. Weitere Informationen finden Sie unter Zugriffspunkt-Kompatibilität mit S3-Vorgänge.

Zugriffspunkt-Operationen

Zugriffspunkt-Operationen sind S3-API-Operationen, die für den `accesspoint` Ressourcentyp ausgeführt werden. Beispiel: `CreateAccessPoint`, `DeleteAccessPoint` und `GetAccessPointPolicy`. S3-Richtlinienaktionen für Zugriffspunkt-Operationen können nur in identitätsbasierten IAM-Richtlinien verwendet werden, nicht in Bucket-Richtlinien oder Zugriffspunkt-Richtlinien. Zugriffspunkt-Operationen erfordern, dass das -`Resource`Element der `accesspoint` ARN im folgenden Beispielformat ist.

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT"
```

Die folgende identitätsbasierte IAM-Richtlinie erteilt die `s3:GetAccessPointPolicy` Berechtigung zum Ausführen der GetAccessPointPolicy API-Operation auf dem S3-Zugriffspunkt *DOC-EXAMPLE-ACCESS-POINT* .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Grant permission to retrieve the access point policy of access
 point DOC-EXAMPLE-ACCESS-POINT",
            "Effect": "Allow",
            "Action": [
            "s3:GetAccessPointPolicy"
            ],
            "Resource": "arn:aws:s3:*:123456789012:access point/DOC-EXAMPLE-ACCESS-
POINT"
        }
    ]
}
```

Wenn Sie Access Points verwenden, um den Zugriff auf Bucket-Operationen zu steuern, finden Sie weitere Informationen unter Bucket-Operationen in Zugriffspunktrichtlinien. Informationen

zur Steuerung des Zugriffs auf Objektoperationen finden Sie unter Objektoperationen in Zugriffspunktrichtlinien. Weitere Informationen zum Konfigurieren von Zugriffspunktrichtlinien finden Sie unter Konfigurieren von IAM-Richtlinien für die Verwendung von Zugriffspunkten.

Operationen für Object Lambda Access Points

Weitere Informationen zum Konfigurieren von Richtlinien für Objekt-Lambda-Zugriffspunkt-Operationen finden Sie unter Konfigurieren von IAM-Richtlinien für Object Lambda Access Points.

Operationen für Multi-Region Access Points

Weitere Informationen zum Konfigurieren von Richtlinien für Multi-Region Access Point-Operationen finden Sie unter Beispielrichtlinien für Multi-Region Access Points.

Batch-Auftragsoperationen

(Batchoperationen) Auftragsoperationen sind S3-API-Operationen, die mit dem Ressourcentyp des Auftrags arbeiten. Beispiel: `DescribeJob` und `CreateJob`. S3-Richtlinienaktionen für Auftragsoperationen können nur in identitätsbasierten IAM-Richtlinien und nicht in Bucket-Richtlinien verwendet werden. Außerdem erfordern Auftragsoperationen, dass das -`Resource`Element in identitätsbasierten IAM-Richtlinien der `job` ARN im folgenden Beispielformat ist.

```
"Resource": "arn:aws:s3:*:123456789012:job/*"
```

Die folgende identitätsbasierte IAM-Richtlinie erteilt die `s3:DescribeJob` Berechtigung zum Ausführen des DescribeJob API-Vorgangs für den S3-Batchoperationenauftrag *DOC-EXAMPLE-JOB* .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow describing the Batch operation job DOC-EXAMPLE-JOB",
            "Effect": "Allow",
            "Action": [
            "s3:DescribeJob"
            ],
            "Resource": "arn:aws:s3:*:123456789012:job/DOC-EXAMPLE-JOB"
        }
    ]
}
```

S3-Storage-Lens-Konfigurationsvorgänge

Weitere Informationen zum Konfigurieren von S3-Storage-Lens-Konfigurationsvorgängen finden Sie unter Berechtigungen für Amazon S3 Storage Lens.

Kontooperationen

Kontooperationen sind S3-API-Operationen, die auf Kontoebene ausgeführt werden. Zum Beispiel `GetPublicAccessBlock` (für -Konto). Das Konto ist kein von Amazon S3 definierter Ressourcentyp. S3-Richtlinienaktionen für Kontooperationen können nur in identitätsbasierten IAM-Richtlinien und nicht in Bucket-Richtlinien verwendet werden. Außerdem erfordern Kontooperationen, dass das -`Resource`Element in identitätsbasierten IAM-Richtlinien ist`"*"`.

Die folgende identitätsbasierte IAM-Richtlinie erteilt die `s3:GetAccountPublicAccessBlock` Berechtigung, den GetPublicAccessBlock API-Vorgang auf Kontoebene auszuführen und die Einstellungen für den öffentlichen Zugriffsblock auf Kontoebene abzurufen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"Allow retrieving the account-level Public Access Block settings",
         "Effect":"Allow",
         "Action":[
            "s3:GetAccountPublicAccessBlock"
         ],
         "Resource":[
            "*"
         ]
       }
    ]
}
```

Amazon-S3-Bedingungsschlüssel

Mit der Sprache der Zugriffsrichtlinie können Sie bei der Erteilung von Berechtigungen Bedingungen angeben. Um Bedingungen dafür festzulegen, wann eine Richtlinie gültig ist, können Sie das optionale Element `Condition` oder den Block `Condition` verwenden. Sie können vordefinierte AWS-weite Schlüssel und Amazon S3-spezifische Schlüssel verwenden, um Bedingungen in einer Amazon S3-Zugriffsrichtlinie anzugeben.

Im Element `Condition` formulieren Sie Ausdrücke, in denen Sie boolesche Operatoren verwenden (gleich, kleiner als usw.), um die Bedingung auf Übereinstimmung mit den Werten in der Anforderung zu prüfen. Wenn Sie einem Benutzer beispielsweise die Berechtigung zum Hochladen eines Objekts erteilen, kann der Bucket-Eigentümer anfordern, dass das Objekt öffentlich lesbar ist, indem er die Bedingung `StringEquals` wie hier gezeigt hinzufügt:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": [
        "arn:aws:s3:::awsexamplebucket1/*"
      ],
      "Condition": {
        "StringEquals": {
          "s3:x-amz-acl": "public-read"
        }
      }
    }
  ]
}
```

Im Beispiel gibt der Block `Condition` die Bedingung `StringEquals` an, die auf das angegebene Schlüssel-Wert-Paar angewendet wird, `"s3:x-amz-acl":["public-read"]`. Es gibt einen Satz vordefinierter Schlüssel, die Sie zum Ausdruck einer Bedingung verwenden können. Das Beispiel verwendet den Bedingungsschlüssel `s3:x-amz-acl`. Diese Bedingung erfordert, dass der Benutzer in jeder PUT Object-Anforderung den Header `x-amz-acl` mit dem Wert `public-read` angibt.

Themen

- AWS-weite Bedingungsschlüssel
- Amazon-S3-spezifische Bedingungsschlüssel
- Beispiele für Amazon-S3-Bedingungsschlüssel

AWS-weite Bedingungsschlüssel

AWS bietet eine Reihe von gemeinsamen Schlüsseln, die von allen - AWS Services unterstützt werden, die Richtlinien unterstützen. Diese Schlüssel werden als AWS-weite Schlüssel bezeichnet und verwenden das Präfix `aws:`. Eine vollständige Liste der AWS-weiten Schlüssel ist im Abschnitt Verfügbare AWS -Schlüssel für Bedingungen des IAM-Benutzerhandbuchs enthalten. Sie können AWS-weite Bedingungsschlüssel in Amazon S3 verwenden. Die folgende Beispiel-Bucket-Richtlinie ermöglicht authentifizierten Benutzern das Verwenden der Aktion `s3:GetObject`, wenn die Anforderung aus einem bestimmten Bereich von IP-Adressen stammt (192.0.2.0.*), sofern die IP-Adresse nicht 192.0.2.188 ist. Im Bedingungsblock sind die Bedingungen `IpAddress` und `NotIpAddress` enthalten, und bei jeder Bedingung ist ein Schlüssel-Wert-Paar zur Auswertung angefügt. Beide Schlüssel-Wert-Paare in diesem Beispiel verwenden den `aws:SourceIp` AWS-weiten Schlüssel.

> **Note**
>
> Die in der Bedingung angegebenen Schlüsselwerte `IPAddress` und `NotIpAddress` verwenden die CIDR-Notation, wie in RFC 4632 beschrieben. Weitere Informationen finden Sie unter http://www.rfc-editor.org/rfc/rfc4632.txt.

```
{
    "Version": "2012-10-17",
    "Id": "S3PolicyId1",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": "*",
            "Action":"s3:GetObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
            "Condition" : {
                "IpAddress" : {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NotIpAddress" : {
                    "aws:SourceIp": "192.0.2.188/32"
                }
            }
        }
```

```
    ]
}
```

Sie können auch andere AWS-weite Bedingungsschlüssel in Amazon S3-Richtlinien verwenden. Beispielsweise können Sie die Bedingungsschlüssel `aws:SourceVpce` und `aws:SourceVpc` in Bucket-Richtlinien für VPC-Endpunkte angeben. Spezifische Beispiele finden Sie unter Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien.

> **Note**
>
> Für einige AWS globale Bedingungsschlüssel werden nur bestimmte Ressourcentypen unterstützt. Prüfen Sie daher, ob Amazon S3 den globalen Bedingungsschlüssel und den Ressourcentyp unterstützt, die Sie verwenden möchten, oder ob Sie stattdessen einen spezifischen Bedingungsschlüssel von Amazon S3 verwenden müssen. Eine vollständige Liste der unterstützten Ressourcentypen und Bedingungsschlüssel für Amazon S3 finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Amazon-S3-spezifische Bedingungsschlüssel

Sie können Amazon-S3-Bedingungsschlüssel mit bestimmten Amazon-S3-Aktionen verwenden. Jeder Bedingungsschlüssel wird dem vom API zugelassenen gleichen Namensanforderungsheader zugeordnet, auf den die Bedingung festgelegt werden kann. Amazon-S3-spezifische Bedingungsschlüssel bestimmen das Verhalten der Anforderungs-Header mit demselben Namen. Eine vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Der Bedingungsschlüssel `s3:x-amz-acl`, den Sie zum Erteilen der Bedingungsberechtigung `s3:PutObject`
verwenden können, definiert das Verhalten des Anforderungs-Headers `x-amz-acl`, den die PUT-Objekt-API unterstützt. Der Bedingungsschlüssel `s3:VersionId`, den Sie für die bedingte Berechtigung für die Bedingung
`s3:GetObjectVersion`
verwenden können, definiert das Verhalten des Abfrageparameters `versionId`, den Sie in einer GET-Objekt-Anforderung setzen.

Die folgende Bucket-Richtlinie erteilt die -`s3:PutObject`Berechtigung für zwei AWS-Konten , wenn die Anforderung den `x-amz-acl` Header enthält, der das Objekt öffentlich lesbar macht. Der

`Condition`-Block verwendet die `StringEquals`-Bedingung und ist mit dem Schlüssel-Wert-Paar `"s3:x-amz-acl":["public-read"` zur Auswertung versehen. Im Schlüssel-Wert-Paar ist `s3:x-amz-acl` ein Amazon S3–spezifischer Schlüssel, wie durch das Präfix `s3:` angezeigt ist.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
         "Sid":"AddCannedAcl",
            "Effect":"Allow",
         "Principal": {
                "AWS": [
                "arn:aws:iam::Account1-ID:root",
    "arn:aws:iam::Account2-ID:root"
    ]
            },
         "Action":"s3:PutObject",
            "Resource": ["arn:aws:s3:::awsexamplebucket1/*"],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl":["public-read"]
                }
            }
        }
    ]
}
```

Important

Nicht alle Bedingungen machen für alle Aktionen auch Sinn. Beispielsweise ist es sinnvoll, die Bedingung `s3:LocationConstraint` für eine Richtlinie einzufügen, die die Amazon-S3-Berechtigung `s3:CreateBucket` erteilt. Es ist jedoch nicht sinnvoll, diese Bedingung in eine Richtlinie aufzunehmen, die die `s3:GetObject`-Genehmigung erteilt. Amazon S3 kann auf semantische Fehler dieses Typs testen, die Amazon-S3-spezifische Bedingungen beinhalten. Wenn Sie jedoch eine Richtlinie für eine(n) IAM-Benutzer oder -Rolle erstellen und eine semantisch ungültige Amazon-S3-Bedingung einfügen, wird kein Fehler gemeldet, weil IAM keine Amazon-S3-Bedingungen validieren kann.

Beispiele für Amazon-S3-Bedingungsschlüssel

Sie können die Sprache der Zugriffsrichtlinie verwenden, um Bedingungen anzugeben, wenn Sie Berechtigungen erteilen. Sie können das optionale Element `Condition` oder den `Condition`-Block verwenden, um anzugeben, unter welchen Bedingungen eine Richtlinie wirksam ist.

Richtlinien, die Amazon-S3-Bedingungsschlüssel für Objekt- und Bucket-Vorgänge verwenden, finden Sie in den folgenden Beispielen. Weitere Informationen über Bedingungsschlüssel finden Sie unter Amazon-S3-Bedingungsschlüssel. Eine vollständige Liste der Amazon S3-Aktionen, Bedingungsschlüssel und -Ressourcen, die Sie in -Richtlinien angeben können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Beispiele – Amazon-S3-Bedingungsschlüssel für Objekt-Vorgänge

Dieser Abschnitt enthält Beispiele, die zeigen, wie Sie Amazon-S3-spezifische Bedingungsschlüssel für Objekt-Vorgänge verwenden können. Eine vollständige Liste der Amazon S3-Aktionen, Bedingungsschlüssel und -Ressourcen, die Sie in -Richtlinien angeben können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Einige der Beispielrichtlinien zeigen, wie Sie Bedingungsschlüssel mit PUT Object-Vorgänge verwenden können. PUT Object-Vorgänge ermöglichen für Access Control List (ACL) spezifische Header, mit denen Sie ACL-basierte Berechtigungen erteilen können. Mit diesen Schlüsseln kann der Bucket-Eigentümer eine Bedingung festlegen, die bestimmte Zugriffsberechtigungen erfordert, wenn der Benutzer ein Objekt hochlädt. Sie können auch ACL-basierte Berechtigungen mit der -PutObjectAcl Operation erteilen. Weitere Informationen finden Sie unter PutObjectAcl in der API-Referenz zu Amazon S3 Amazon Simple Storage Service. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht.

Themen

- Beispiel 1: Erteilen von s3:PutObject permission mit einer Bedingung, die erfordert, dass der Bucket-Eigentümer die volle Kontrolle erhält
- Beispiel 2: Erteilen einer s3:PutObject -Berechtigung, die die Speicherung von Objekten mit serverseitiger Verschlüsselung erfordert
- Beispiel 3: Erteilen der s3:PutObject -Berechtigung zum Kopieren von Objekten mit einer Einschränkung für die Kopierquelle
- Beispiel 4: Zugriff auf eine bestimmte Version eines Objekts gewähren

- Beispiel 5: Objekt-Uploads auf Objekte mit einer bestimmten Speicherklasse beschränken
- Beispiel 6: Erteilen von Berechtigungen basierend auf Objekt-Markierungen
- Beispiel 7: Beschränken des Zugriffs durch die AWS-Konto ID des Bucket-Eigentümers
- Beispiel 8: Erfordern einer minimalen TLS-Version

Beispiel 1: Erteilen von s3:PutObject permission mit einer Bedingung, die erfordert, dass der Bucket-Eigentümer die volle Kontrolle erhält

Die Operation PUT Object erlaubt Access Control List (ACL)-spezifische Header, mit denen Sie ACL-spezifische Berechtigungen erteilen können. Mit diesen Schlüsseln kann der Bucket-Eigentümer eine Bedingung festlegen, die bestimmte Zugriffsberechtigungen erfordert, wenn der Benutzer ein Objekt hochlädt.

Angenommen, Konto A besitzt einen Bucket und der Kontoadministrator möchte Akua, einem Benutzer in Konto B, Berechtigungen zum Hochladen von Objekten erteilen. Standardmäßig gehören Objekte, die Akua hochlädt, Konto B, und Konto A hat keine Berechtigungen für diese Objekte. Da der Bucket-Eigentümer die Rechnungen zahlt, benötigt er volle Berechtigungen für die Objekte, die Akua hochlädt. Der Administrator von Konto A kann dies tun, indem er Akua die `s3:PutObject` Berechtigung erteilt, mit der Bedingung, dass die Anforderung ACL-spezifische Header enthält, die entweder explizit die volle Berechtigung erteilen oder eine vordefinierte ACL verwenden. Weitere Informationen finden Sie unter PUT Object.

Erzwingen des - x-amz-full-control Headers

Sie können in der Anforderung den Header `x-amz-full-control` mit vollständiger Kontrollberechtigung für den Bucket-Eigentümer anfordern. Die folgende Bucket-Richtlinie erteilt dem Benutzer Akua die `s3:PutObject` Berechtigung mit einer Bedingung, die den `s3:x-amz-grant-full-control` Bedingungsschlüssel verwendet, der erfordert, dass die Anforderung den `x-amz-full-control` Header enthält.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountB-ID:user/Akua"
      },
```

```
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::awsexamplebucket1/*",
      "Condition": {
        "StringEquals": {
          "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
        }
      }
    }
  ]
}
```

> **Note**
>
> In diesem Beispiel geht es um die kontoübergreifende Berechtigung. Wenn Akua (der die Berechtigung erhält) jedoch zu dem gehört AWS-Konto , dem der Bucket gehört, ist diese bedingte Berechtigung nicht erforderlich. Dies liegt daran, dass das übergeordnete Konto, zu dem Akua gehört, Eigentümer von Objekten ist, die der Benutzer hochlädt.

Hinzufügen einer expliziten Zugriffsverweigerung

Die vorangehende Bucket-Richtlinie gewährt dem Benutzer Akua in Konto B eine bedingte Berechtigung. Während diese Richtlinie in Kraft ist, ist es möglich, dass Akua dieselbe Berechtigung ohne Bedingung über eine andere Richtlinie erhält. Beispielsweise kann Akua zu einer Gruppe gehören, und Sie erteilen der Gruppe die `s3:PutObject` Berechtigung ohne Bedingung. Um solche Berechtigungslücken zu vermeiden, können Sie eine strengere Zugriffsrichtlinie schreiben, indem Sie eine explizite Zugriffsverweigerung hinzufügen. In diesem Beispiel verweigern Sie dem Benutzer Akua explizit die Upload-Berechtigung, wenn er nicht die erforderlichen Header in die Anforderung aufnimmt, die dem Bucket-Eigentümer vollständige Berechtigungen erteilt. Eine explizite Verweigerung ersetzt immer eine anderswo erteilte Erlaubnis. Im Folgenden finden Sie das geänderte Zugriffsrichtlinienbeispiel mit hinzugefügter expliziter Ablehnung.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
```

```
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
                }
            }
        },
        {
            "Sid": "statement2",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
                }
            }
        }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Wenn Sie zwei haben AWS-Konten, können Sie die Richtlinie mit der AWS Command Line Interface (AWS CLI) testen. Sie fügen die Richtlinie an und verwenden die Anmeldeinformationen von Akua, um die Berechtigung mit dem folgenden AWS CLI `put-object` Befehl zu testen. Sie geben die Anmeldeinformationen von Akua an, indem Sie den `--profile` Parameter hinzufügen. Sie erteilen dem Bucket-Eigentümer die volle Kontrolle, indem Sie den Parameter `--grant-full-control` hinzufügen. Weitere Informationen zum Einrichten und Verwenden der finden Sie AWS CLI unter Entwickeln mit Amazon S3 über die AWS CLI.

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body c:\HappyFace.jpg
 --grant-full-control id="AccountA-CanonicalUserID" --profile AccountBUserProfile
```

Erzwingen des - x-amz-acl Headers

Sie können den Header `x-amz-acl` mit einer vordefinierten ACL anfordern, die dem Bucket-Eigentümer die vollständige Kontrollberechtigung erteilt. Um den Header `x-amz-acl` in der Anforderung erforderlich zu machen, können Sie das Schlüssel-Wert-Paar im Block `Condition` ersetzen und den Bedingungsschlüssel `s3:x-amz-acl` wie im folgenden Beispiel dargestellt angeben.

```
"Condition": {
    "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control"
    }
}
```

Um die Berechtigung mit der zu testen AWS CLI, geben Sie den Parameter an`--acl`. Der fügt AWS CLI dann den `x-amz-acl` Header hinzu, wenn er die Anforderung sendet.

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body c:\HappyFace.jpg
 --acl "bucket-owner-full-control" --profile AccountBadmin
```

Beispiel 2: Erteilen einer s3:PutObject -Berechtigung, die die Speicherung von Objekten mit serverseitiger Verschlüsselung erfordert

Angenommen, Konto A besitzt einen Bucket. Der Konto-Administrator möchte Jane, einer Benutzerin in Konto A, die Berechtigung zum Hochladen von Objekten mit der Bedingung erteilen, dass Jane immer serverseitige Verschlüsselung anfordert, damit Amazon S3 die Objekte verschlüsselt speichert. Der Administrator von Konto A kann für diesen Zweck wie gezeigt mit dem Bedingungsschlüssel `s3:x-amz-server-side-encryption` arbeiten. Das Schlüssel-Wert-Paar im Block `Condition` gibt den Schlüssel `s3:x-amz-server-side-encryption` an.

```
"Condition": {
     "StringNotEquals": {
         "s3:x-amz-server-side-encryption": "AES256"
     }}
```

Wenn Sie die Berechtigung mit der testen AWS CLI, müssen Sie den erforderlichen Parameter mit dem `--server-side-encryption` Parameter hinzufügen.

```
aws s3api put-object --bucket example1bucket --key HappyFace.jpg --body c:
\HappyFace.jpg --server-side-encryption "AES256" --profile AccountBadmin
```

Beispiel 3: Erteilen der s3:PutObject -Berechtigung zum Kopieren von Objekten mit einer Einschränkung für die Kopierquelle

Wenn Sie in der PUT Object-Anforderung ein Quellobjekt angeben, handelt es sich um eine Kopieroperation (siehe PUT Object - Copy). Dementsprechend kann der Bucket-Eigentümer einem Benutzer die Berechtigung erteilen, Objekte mit Einschränkungen in Bezug auf die Quelle zu kopieren, z. B.:

- das Kopieren von Objekten nur aus dem `sourcebucket`-Bucket erlauben.
- das Kopieren von Objekten aus dem Quell-Bucket und nur den Objekten, deren Schlüsselnamen-Präfix mit public/f beginnt, erlauben (z. B. sourcebucket/public/*)
- das Kopieren nur eines bestimmten Objekts aus dem Quell-Bucket erlauben (z. B. sourcebucket/example.jpg).

Die folgende Bucket-Richtlinie erteilt dem Benutzer (Akua) die `s3:PutObject` Berechtigung. Sie erlaubt ihm, nur Objekte mit der Bedingung zu kopieren, dass die Anforderung den Header `s3:x-amz-copy-source` enthält und der Header-Wert das Präfix des Schlüsselnamens `/awsexamplebucket1/public/*` angibt.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Sid": "cross-account permission to user in your own account",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Akua"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*"
        },
        {
            "Sid": "Deny your user permission to upload object if copy source is not /
bucket/folder",
            "Effect": "Deny",
            "Principal": {
```

```
                "AWS": "arn:aws:iam::123456789012:user/Akua"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
            "Condition": {
                "StringNotLike": {
                    "s3:x-amz-copy-source": "awsexamplebucket1/public/*"
                }
            }
        }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Berechtigung mit dem AWS CLI `copy-object` Befehl testen. Sie geben die Quelle an, indem Sie den Parameter `--copy-source` hinzufügen. Das Schlüsselnamenpräfix muss dem Präfix entsprechen, das in der Richtlinie zulässig ist. Sie müssen dem Benutzer Akua-Anmeldeinformationen mit dem `---profile`Parameter bereitstellen. Weitere Informationen zum Einrichten der finden Sie AWS CLI unter Entwickeln mit Amazon S3 über die AWS CLI.

```
aws s3api copy-object --bucket awsexamplebucket1 --key HappyFace.jpg
--copy-source examplebucket/public/PublicHappyFace1.jpg --profile AccountAAkua
```

Erteilen der Berechtigung, nur ein bestimmtes Objekt zu kopieren

Die vorherige Richtlinie verwendet die Bedingung `StringNotLike`. Um nur das Kopieren eines bestimmten Objekts zu erlauben, müssen Sie die Bedingung von `StringNotLike` in `StringNotEquals` ändern und dann den genauen Objektschlüssel wie gezeigt angeben.

```
"Condition": {
       "StringNotEquals": {
           "s3:x-amz-copy-source": "awsexamplebucket1/public/PublicHappyFace1.jpg"
       }
}
```

Beispiel 4: Zugriff auf eine bestimmte Version eines Objekts gewähren

Angenommen, Konto A besitzt einen versionsaktivierten Bucket. Der Bucket beinhaltet mehrere Versionen des Objekts `HappyFace.jpg`. Der Kontoadministrator möchte nun seinem Benutzer Akua die Berechtigung erteilen, nur eine bestimmte Version des Objekts abzurufen. Der Kontoadministrator

kann dies erreichen, indem er Akua die `s3:GetObjectVersion` Berechtigung bedingt erteilt, wie unten gezeigt. Das Schlüssel-Wert-Paar im Block `Condition` gibt den Bedingungsschlüssel `s3:VersionId` an. In diesem Fall muss Akua die genaue Objektversions-ID kennen, um das Objekt abzurufen.

Weitere Informationen finden Sie unter GetObject in der API-Referenz zu Amazon Simple Storage Service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Akua"
            },
            "Action": "s3:GetObjectVersion",
            "Resource": "arn:aws:s3:::examplebucketversionenabled/HappyFace.jpg"
        },
        {
            "Sid": "statement2",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Akua"
            },
            "Action": "s3:GetObjectVersion",
            "Resource": "arn:aws:s3:::examplebucketversionenabled/HappyFace.jpg",
            "Condition": {
                "StringNotEquals": {
                    "s3:VersionId": "AaaHbAQitwiL_h47_44lRO2DDfLlBO5e"
                }
            }
        }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Berechtigungen mit dem AWS CLI `get-object` Befehl mit dem `--version-id` Parameter testen, der die spezifische Objektversion identifiziert. Der Befehl ruft das Objekt ab und speichert es in der Datei `OutputFile.jpg`.

```
aws s3api get-object --bucket examplebucketversionenabled --key HappyFace.jpg
 OutputFile.jpg --version-id AaaHbAQitwiL_h47_44lRO2DDfLlBO5e --profile AccountAAkua
```

Beispiel 5: Objekt-Uploads auf Objekte mit einer bestimmten Speicherklasse beschränken

Angenommen, Konto A, dargestellt durch Konto-ID 123456789012, besitzt einen Bucket. Der Kontoadministrator möchte Akua, einen Benutzer in Konto A, einschränken, um nur Objekte in den Bucket hochladen zu können, die in der `STANDARD_IA` Speicherklasse gespeichert sind. Um das Hochladen von Objekten auf eine bestimmte Speicherklasse einzuschränken, kann der Administrator von Konto A den Bedingungsschlüssel `s3:x-amz-storage-class` verwenden, wie in der folgenden Bucket-Beispielrichtlinie gezeigt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/Akua"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
      "Condition": {
        "StringEquals": {
          "s3:x-amz-storage-class": [
            "STANDARD_IA"
          ]
        }
      }
    }
  ]
}
```

Beispiel 6: Erteilen von Berechtigungen basierend auf Objekt-Markierungen

Beispiele für die Verwendung von Objektmarkierungs-Bedingungsschlüsseln mit Amazon-S3-Vorgänge finden Sie unter Markierungs- und Zugriffskontrollrichtlinien.

Beispiel 7: Beschränken des Zugriffs durch die AWS-Konto ID des Bucket-Eigentümers

Sie können entweder den Schlüssel `aws:ResourceAccount` oder den Schlüssel `s3:ResourceAccount` verwenden, um Endpunktrichtlinien für IAM oder Virtual Private Cloud (VPC) zu schreiben, die den Benutzer- oder Anwendungszugriff auf die Amazon-S3-Buckets einschränken, die einer bestimmten AWS-Konto -ID gehören. Sie können diesen Bedingungsschlüssel verwenden, wenn Sie Clients in Ihrer VPC daran hindern möchten, auf Buckets zuzugreifen, die Sie nicht besitzen.

Beachten Sie jedoch, dass einige - AWS Services auf den Zugriff auf AWS verwaltete Buckets angewiesen sind. Daher kann es auch den Zugriff auf diese Ressourcen beeinflussen, wenn Sie den Schlüssel `aws:ResourceAccount` oder `s3:ResourceAccount` in Ihrer IAM-Richtlinie verwenden.

Weitere Informationen und Beispiele finden Sie in den folgenden Ressourcen:

- Beschränken des Zugriffs auf Buckets in einem bestimmten AWS-Konto im AWS PrivateLink - Handbuch
- Beschränken des Zugriffs auf Buckets, die Amazon ECR verwendet im Amazon-ECR-Leitfaden
- Bereitstellen des erforderlichen Zugriffs auf Systems Manager für AWS verwaltete Amazon S3-Buckets im -AWS Systems Manager Handbuch
- Beschränken des Zugriffs auf Amazon-S3-Buckets im Besitz bestimmter AWS-Konten im AWS Storage Blog

Beispiel 8: Erfordern einer minimalen TLS-Version

Sie können den Schlüssel s3:TlsVersion condition verwenden, um IAM-, Virtual Private Cloud Endpoint (VPCE)- oder Bucket-Richtlinien zu schreiben, die den Benutzer- oder Anwendungszugriff auf Amazon S3-Buckets basierend auf der vom Client verwendeten TLS-Version einschränken. Sie können diesen Bedingungsschlüssel verwenden, um Richtlinien zu schreiben, die eine minimale TLS-Version erfordern.

Example

Diese Bucket-Beispielrichtlinie lehnt PutObject Anforderungen von Clients ab, die eine TLS-Version unter 1.2 haben, z. B. 1.1 oder 1.0.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ],
            "Condition": {
                "NumericLessThan": {
                    "s3:TlsVersion": 1.2
                }
            }
        }
    ]
}
```

Example

Diese Beispiel-Bucket-Richtlinie erlaubt PutObject Anforderungen von Clients mit einer TLS-Version höher als 1.1, z. B. 1.2, 1.3 oder höher.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ],
            "Condition": {
                "NumericGreaterThan": {
                    "s3:TlsVersion": 1.1
                }
            }
        }
```

```
    ]
}
```

Beispiele – Amazon-S3-Bedingungsschlüssel für Bucket-Vorgänge

Dieser Abschnitt enthält Beispielrichtlinien, die zeigen, wie Sie Amazon-S3-spezifische Bedingungsschlüssel für Bucket-Vorgänge verwenden können.

Themen

- Beispiel 1: Erteilen einer Benutzerberechtigung zum Erstellen von Buckets nur in einer bestimmten Region
- Beispiel 2: Abrufen einer Liste von Objekten in einem Bucket mit einem bestimmten Präfix
- Beispiel 3: Festlegen der maximalen Anzahl von Schlüsseln

Beispiel 1: Erteilen einer Benutzerberechtigung zum Erstellen von Buckets nur in einer bestimmten Region

Angenommen, ein AWS-Konto Administrator möchte seinem Benutzer (Akua) die Berechtigung erteilen, einen Bucket nur in der Region Südamerika (São Paulo) zu erstellen. Der Kontoadministrator kann die folgende Benutzerrichtlinie anfügen, welche die Berechtigung `s3:CreateBucket` mit der angegebenen Bedingung gewährt. Das Schlüssel-Wert-Paar im Block `Condition` gibt den Schlüssel `s3:LocationConstraint` und die Region `sa-east-1` als seinen Wert an.

> **Note**
>
> In diesem Beispiel erteilt der Bucket-Eigentümer einem seiner Benutzer die Berechtigung, daher kann entweder eine Bucket-Richtlinie oder eine Benutzerrichtlinie verwendet werden. Dieses Beispiel zeigt eine Benutzerrichtlinie.

Die Liste der Amazon-S3-Regionen finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
         "Sid":"statement1",
         "Effect":"Allow",
         "Action": "s3:CreateBucket",
         "Resource": "arn:aws:s3:::*",
         "Condition": {
             "StringLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
       }
    ]
}
```

Hinzufügen einer expliziten Zugriffsverweigerung

Die vorangegangene Richtlinie hindert den Benutzer an der Erstellung eines Buckets in einer anderen Region als `sa-east-1`. Einige andere Richtlinien gewähren diesem Benutzer möglicherweise jedoch die Berechtigung, Buckets in einer anderen Region zu erstellen. Wenn der Benutzer beispielsweise zu einer Gruppe gehört, ist der Gruppe möglicherweise eine Richtlinie angefügt, die alle Benutzer der Gruppe zum Erstellen von Buckets in einer anderen Region berechtigt. Um sicherzustellen, dass der Benutzer keine Berechtigung zum Erstellen von Buckets in einer anderen Region erhält, können Sie in der oben gezeigten Richtlinie eine explizite Anweisung zur Ablehnung hinzufügen.

Die Anweisung `Deny` verwendet die Bedingung `StringNotLike`. Das bedeutet, dass, eine Anforderung zum Erstellen eines Buckets abgelehnt wird, wenn die Standorteinschränkung nicht `sa-east-1` ist. Die explizite Verweigerung erlaubt dem Benutzer nicht, einen Bucket in einer anderen Region zu erstellen, unabhängig davon, welche andere Berechtigung der Benutzer erhält. Die folgende Richtlinie enthält eine explizite Zugriffsverweigerungsanweisung.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action": "s3:CreateBucket",
         "Resource": "arn:aws:s3:::*",
         "Condition": {
             "StringLike": {
                 "s3:LocationConstraint": "sa-east-1"
```

```
            }
         }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
         "Action": "s3:CreateBucket",
         "Resource": "arn:aws:s3:::*",
         "Condition": {
             "StringNotLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
       }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Richtlinie mit dem folgenden `create-bucket` AWS CLI Befehl testen. In diesem Beispiel wird die Datei `bucketconfig.txt` verwendet, um die Standortbeschränkung anzugeben. Merken Sie sich den Windows-Pfad der Datei. Sie müssen den Bucket-Namen und -Pfad entsprechend aktualisieren. Sie müssen die Anmeldeinformationen des Benutzers bereitstellen, indem Sie den Parameter `--profile` hinzufügen. Weitere Informationen zum Einrichten und Verwenden der finden Sie AWS CLI unter Entwickeln mit Amazon S3 über die AWS CLI.

```
aws s3api create-bucket --bucket examplebucket --profile AccountAAkua --create-bucket-
configuration file://c:/Users/someUser/bucketconfig.txt
```

Die Datei `bucketconfig.txt` gibt die Konfiguration wie folgt an.

```
{"LocationConstraint": "sa-east-1"}
```

Beispiel 2: Abrufen einer Liste von Objekten in einem Bucket mit einem bestimmten Präfix

Sie können den `s3:prefix` Bedingungsschlüssel verwenden, um die Antwort der GET Bucket (ListObjects)-API auf Schlüsselnamen mit einem bestimmten Präfix zu beschränken. Wenn Sie der Bucket-Eigentümer sind, können Sie einen Benutzer beschränken, den Inhalt eines bestimmten Präfixes im Bucket aufzulisten. Dieser Bedingungsschlüssel ist nützlich, wenn Objekte im Bucket nach Schlüsselnamenpräfixen organisiert sind. Die Amazon-S3-Konsole

verwendet Schlüsselnamenpräfixe zum Anzeigen von Ordnerkonzepten. Nur die Konsole unterstützt das Ordnerkonzept. Die Amazon-S3-API unterstützt ausschließlich Buckets und Objekte. Weitere Informationen zur Verwendung von Präfixen und Trennzeichen zum Filtern von Zugriffsberechtigungen finden Sie unter Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien.

Wenn es beispielsweise zwei Objekte mit den Schlüsselnamen `public/object1.jpg` und `public/object2.jpg` gibt, zeigt die Konsole die Objekte unter dem Ordner `public` an. In der Amazon-S3-API sind dies Objekte mit Präfixen, nicht Objekte in Ordnern. Wenn Sie jedoch in der Amazon-S3-API die Objektschlüssel mithilfe solcher Präfixe organisieren, können Sie die Berechtigung `s3:ListBucket` mit der Bedingung `s3:prefix` erteilen, die dem Benutzer das Abrufen einer Liste mit Schlüsselnamen mit diesen spezifischen Präfixen ermöglicht.

In diesem Beispiel sind der Bucket-Eigentümer und das übergeordnete Konto, zu dem der Benutzer gehört, dieselben. Der Bucket-Eigentümer kann also entweder eine Bucket-Richtlinie oder eine Benutzerrichtlinie verwenden. Weitere Informationen zu anderen Bedingungsschlüsseln, die Sie mit der GET Bucket (ListObjects)-API verwenden können, finden Sie unter ListObjects.

Richtlinie für Benutzer:

Die folgende Benutzerrichtlinie erteilt die `s3:ListBucket`-Berechtigung (siehe GET Bucket (List Objects)) mit der Bedingung, dass der Benutzer in der Anforderung das `prefix` mit dem Wert `projects` anführt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action": "s3:ListBucket",
         "Resource":"arn:aws:s3:::awsexamplebucket1",
         "Condition" : {
             "StringEquals" : {
                 "s3:prefix": "projects"
             }
          }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
```

```
        "Action": "s3:ListBucket",
        "Resource": "arn:aws:s3:::awsexamplebucket1",
        "Condition" : {
            "StringNotEquals" : {
                "s3:prefix": "projects"
            }
         }
       }
    ]
}
```

Die Bedingung schränkt den Benutzer ein, nur Objektschlüssel mit dem Präfix `projects` auflisten zu können. Die hinzugefügte explizite Verweigerung verhindert, dass die Anforderung des Benutzers auf Auflistung von Schlüsseln mit irgendeinem anderen Präfix verweigert wird, unabhängig davon, welche anderen Berechtigungen der Benutzer gegebenenfalls hat. z. B. ist es möglich, dass der Benutzer die Berechtigung erhält, Objektschlüssel ohne Einschränkung aufzulisten, entweder durch Aktualisierungen der vorherigen Benutzerrichtlinie oder durch eine Bucket-Richtlinie. Da jedoch die explizite Zugriffsverweigerung stets Vorrang hat, wird die Benutzeranforderung zum Auflisten anderer Schlüssel, die nicht das Präfix `projects` enthalten, abgelehnt.

Bucket-Richtlinie

Wenn Sie der oben gezeigten Richtlinie das Element `Principal` hinzufügen, das den Benutzer identifiziert, erhalten Sie eine Bucket-Richtlinie wie gezeigt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/bucket-owner"
         },
         "Action":  "s3:ListBucket",
         "Resource": "arn:aws:s3:::awsexamplebucket1",
         "Condition" : {
             "StringEquals" : {
                 "s3:prefix": "projects"
             }
          }
       },
```

```
      {
         "Sid":"statement2",
         "Effect":"Deny",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/bucket-owner"
         },
         "Action": "s3:ListBucket",
         "Resource": "arn:aws:s3:::awsexamplebucket1",
         "Condition" : {
             "StringNotEquals" : {
                 "s3:prefix": "projects"
             }
          }
       }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Richtlinie mit dem folgenden `list-object` AWS CLI Befehl testen. Im Befehl geben Sie Benutzeranmeldeinformationen mit dem Parameter `--profile` an. Weitere Informationen zum Einrichten und Verwenden der finden Sie AWS CLI unter Entwickeln mit Amazon S3 über die AWS CLI.

```
aws s3api list-objects --bucket awsexamplebucket1 --prefix examplefolder --profile
 AccountAAkua
```

Wenn der Bucket versionsaktiviert ist, müssen Sie die Berechtigung `s3:ListBucketVersions` in der vorherigen Richtlinie erteilen, um die Objekte im Bucket auflisten zu können, und nicht die Berechtigung `s3:ListBucket`. Diese Berechtigung unterstützt auch den `s3:prefix`-Bedingungsschlüssel.

Beispiel 3: Festlegen der maximalen Anzahl von Schlüsseln

Sie können den `s3:max-keys` Bedingungsschlüssel verwenden, um die maximale Anzahl von Schlüsseln festzulegen, die der Anforderer in einem GET Bucket (ListObjects) oder einer ListObjectVersions Anforderung zurückgeben kann. Standardmäßig gibt die API bis zu 1000 Schlüssel zurück. Die Liste der numerischen Bedingungsoperatoren, die Sie mit `s3:max-keys` und begleitenden Beispielen verwenden können, finden Sie unter Numerische Bedingungsoperatoren im IAM-Benutzerhandbuch.

Verwenden von Bucket-Richtlinien

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Amazon-S3-Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Diese Berechtigungen gelten nicht für Objekte, die anderen gehören AWS-Konten.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, und ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Der Bucket-Eigentümer besitzt alle Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Richtlinien.

Bucket-Richtlinien verwenden JSON-basierte IAM-Richtliniensprache. Sie können Bucket-Richtlinien verwenden, um Berechtigungen für die Objekte in einem Bucket hinzuzufügen oder zu verweigern. Bucket-Richtlinien können Anforderungen basierend auf den Elementen in der Richtlinie erlauben oder verweigern. Diese Elemente umfassen den Anforderer, S3-Aktionen, Ressourcen und Aspekte oder Bedingungen der Anforderung (z. B. die IP-Adresse, die für die Anforderung verwendet wird).

Sie können z. B. eine Bucket-Richtlinie erstellen, die Folgendes bewirkt:

- Erteilt anderen Konten kontoübergreifende Berechtigungen für das Hochladen von Objekten in Ihren S3-Bucket
- Stellt sicher, dass Sie als Bucket-Eigentümer die volle Kontrolle über die hochgeladenen Objekte haben

Weitere Informationen finden Sie unter Beispiele für Bucket-Richtlinien.

Die Themen in diesem Abschnitt enthalten Beispiele und zeigen Ihnen, wie Sie in der S3-Konsole eine Bucket-Richtlinie hinzufügen können. Weitere Informationen zu IAM-Benutzerrichtlinien finden Sie unter Verwenden von IAM-Benutzer- und -rollenrichtlinien. Weitere Informationen zur Sprache der Bucket-Richtlinie finden Sie unter Richtlinien und Berechtigungen in Amazon S3.

Themen

- Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole
- Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien

- Beispiele für Bucket-Richtlinien

Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole

Sie können die den AWS -Richtliniengenerator und die Amazon-S3-Konsole verwenden, um eine neue Bucket-Richtlinie hinzuzufügen oder eine vorhandene zu bearbeiten. Eine Bucket-Richtlinie ist eine ressourcenbasierte AWS Identity and Access Management (IAM)-Richtlinie. Sie fügen einem Bucket eine Bucket-Richtlinie hinzu, um anderen AWS-Konten oder IAM-Benutzern Zugriffsberechtigungen für den Bucket und die darin enthaltenen Objekte zu erteilen. Objektberechtigungen gelten nur für die Objekte, die der Bucket-Eigentümer erstellt. Weitere Informationen zu Bucket-Richtlinien finden Sie unter Übersicht über die Verwaltung von Zugriffsberechtigungen.

Beheben Sie Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge von AWS Identity and Access Management Access Analyzer bevor Sie Ihre Richtlinie speichern. IAM Access Analyzer führt Richtlinienprüfungen durch, um Ihre Richtlinie anhand der IAM-Richtliniengrammatik und der bewährten Methoden zu validieren. Diese Prüfungen generieren Ergebnisse und bieten umsetzbare Empfehlungen, die Sie beim Erstellen von Richtlinien unterstützen, die funktionsfähig sind und den bewährten Methoden für Sicherheit entsprechen. Weitere Informationen zum Validieren von Richtlinien mit IAM Access Analyzer finden Sie unter Validierung der IAM-Access-Analyzer-Richtlinien im IAM-Benutzerhandbuch. Eine Liste der Warnungen, Fehler und Vorschläge, die von IAM Access Analyzer zurückgegeben werden, finden Sie unter IAM-Access-Analyzer-Richtlinienprüfungsreferenz.

Hinweise zur Behebung von Fehlern mit einer Richtlinie finden Sie unter Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3.

Eine Bucket-Richtlinie erstellen oder bearbeiten

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie eine Bucket-Richtlinie erstellen wollen oder dessen Bucket-Richtlinie Sie bearbeiten wollen.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Wählen Sie unter Bucket-Richtlinie Bearbeiten aus. Die Seite Edit bucket policy (Bucket-Richtlinie bearbeiten) wird angezeigt.

6. Führen Sie auf der Seite Edit bucket policy (Bucket-Richtlinie bearbeiten) einen der folgenden Schritte aus:

   - Wenn Sie Beispiele für Bucket-Richtlinien im Amazon-S3-Benutzerhandbuch sehen möchten, wählen Sie Policy examples (Richtlinienbeispiele) aus.
   - Um eine Richtlinie automatisch zu generieren oder den JSON im Abschnitt Policy (Richtlinie) zu bearbeiten, wählen Sie Policy generator (Richtliniengenerator) aus.

   Wenn Sie Richtliniengenerator auswählen, wird der AWS Richtliniengenerator in einem neuen Fenster geöffnet.

   a. Wählen Sie auf der Seite AWS Policy Generator (Richtliniengenerator) unter Select Type of Policy (Richtlinientyp auswählen) die Option S3 Bucket Policy (S3-Bucket-Richtlinie) aus.

   b. Fügen Sie eine Anweisung hinzu, indem Sie die Informationen in die bereitgestellten Felder eingeben, und wählen Sie dann Anweisung hinzufügen. Wiederholen Sie diesen Vorgang für so viele Anweisungen, wie Sie hinzufügen möchten. Weitere Informationen zu diesen Feldern finden Sie in der Referenz zu den IAM-JSON-Richtlinienelementen im IAM-Benutzerhandbuch.

      > **Note**
      >
      > Der Einfachheit halber zeigt die Seite Edit bucket policy (Bucket-Richtlinie bearbeiten) den Bucket ARN (Bucket-ARN) (Amazon-Ressourcenname) des aktuellen Buckets über dem Textfeld Policy (Richtlinie) an. Sie können diesen ARN zur Verwendung in den Anweisungen auf der Seite AWS -Richtliniengenerator kopieren.

   c. Wenn Sie mit dem Hinzufügen von Anweisungen fertig sind, wählen Sie Generieren von Richtlinien.

   d. Kopieren Sie den generierten Richtlinientext, wählen Sie Schließen und kehren Sie zur Seite Bucket-Richtlinie bearbeiten in der Amazon-S3-Konsole zurück.

7. Bearbeiten Sie im Feld Richtlinie die vorhandene Richtlinie oder fügen Sie die Bucket-Richtlinie aus dem AWS Richtliniengenerator ein. Beheben Sie Sicherheitswarnungen, Fehler, allgemeine Warnungen und Vorschläge bevor Sie Ihre Richtlinie speichern.

> **Note**
>
> Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt.

8. (Optional) Wählen Sie unten rechts Preview external access (Vorschau des externen Zugriffs) aus, um eine Vorschau anzuzeigen, wie sich Ihre neue Richtlinie auf den öffentlichen und kontoübergreifenden Zugriff auf Ihre Ressource auswirkt. Bevor Sie Ihre Richtlinie speichern, können Sie überprüfen, ob sie neue IAM-Access-Analyzer-Ergebnisse einführt oder vorhandene Ergebnisse löst. Wenn Sie keinen aktiven Analyzer sehen, wählen Sie Go to Access Analyzer (Zu Access Analyzer wechseln) aus, um einen Account Analyzer in IAM Access Analyzer zu erstellen. Weitere Informationen finden Sie unter Zugriffsvorschau im IAM-Benutzerhandbuch.
9. Wählen Sie Save changes (Änderungen speichern) aus, wodurch Sie zur Registerkarte Permissions (Berechtigungen) zurückkehren.

Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien

Sie können Amazon-S3-Bucket-Richtlinien verwenden, um den Zugriff auf Buckets von bestimmten Virtual Private Cloud (VPC)-Endpunkten oder bestimmten VPCs aus zu steuern. Dieser Abschnitt enthält Beispiele für Bucket-Richtlinien, die für die Steuerung des Zugriffs auf Amazon-S3-Buckets von VPC-Endpunkten aus verwendet werden können. Informationen zum Einrichten von VPC-Endpunkten finden Sie unter VPC-Endpunkte im VPC-Benutzerhandbuch.

Mit der VPC können Sie - AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk starten. Ein VPC-Endpunkt ermöglicht es Ihnen, eine private Verbindung zwischen Ihrer VPC und einem anderen - AWS Service herzustellen, ohne dass ein Zugriff über das Internet, über eine VPN-Verbindung, über eine NAT-Instance oder über erforderlich ist AWS Direct Connect.

Ein Amazon VPC-Endpunkt für Amazon S3 ist eine logische Entität innerhalb einer VPC, die eine Verbindung nur zu Amazon S3 zulässt. Der VPC-Endpunkt leitet Anfragen an Amazon S3 weiter und die Antworten zurück an die VPC. VPC-Endpunkte steuern nur, wie Anfragen weitergeleitet werden. Die öffentlichen Endpunkte und DNS-Namen von Amazon S3 verwenden weiterhin VPC-Endpunkte. Wichtige Informationen zur Verwendung von VPC-Endpunkten mit Amazon S3 finden Sie unter Gateway VPC-Endpunkte und Endpunkte für Amazon S3 im VPC-Benutzerhandbuch.

VPC-Endpunkte für Amazon S3 bieten zwei Möglichkeiten, den Zugriff auf Ihre Amazon-S3-Daten zu steuern:

- Sie können steuern, welche Anfragen, Benutzer oder Gruppen durch einen spezifischen VPC-Endpunkt erlaubt sind. Informationen zu dieser Art der Zugriffskontrolle finden Sie unter Kontrolle des Zugriffs auf Services mit VPC-Endpunkten im VPC-Benutzerhandbuch.
- Sie können steuern, welche VPCs oder VPC-Endpunkte Zugriff auf Ihre Buckets haben, indem Sie Amazon-S3-Bucket-Richtlinien verwenden. Beispiele für diese Art Zugriffssteuerung durch Bucket-Richtlinienfinden Sie in den folgenden Themen zur Zugriffsbeschränkung.

Themen

- Beschränken des Zugriffs auf einen bestimmten VPC-Endpunkt
- Beschränkung des Zugriffs auf eine bestimmte VPC

> **Important**
>
> Wenn Sie die in diesem Abschnitt beschriebenen Amazon-S3-Bucket-Richtlinien für VPC-Endpunkte anwenden, können Sie Ihren Zugriff auf den Bucket blockieren, ohne dies zu beabsichtigen. Bucket-Berechtigungen, die den Bucket-Zugriff auf Verbindungen, die von Ihrem VPC-Endpunkt ausgehen, gezielt einschränken sollen, können alle Verbindungen zum Bucket blockieren. Informationen zur Behebung dieses Problems finden Sie unter My bucket policy has the wrong VPC or VPC endpoint ID (Meine Bucket-Richtlinie hat die falsche VPC- oder VPC-Endpunkt-ID). Wie kann ich die Richtlinie so ändern, dass ich auf den Bucket zugreifen kann? im AWS Support Knowledge Center.

Beschränken des Zugriffs auf einen bestimmten VPC-Endpunkt

Nachfolgend finden Sie ein Beispiel für eine Amazon-S3-Bucket-Richtlinie, die den Zugriff auf einen bestimmten Bucket einschränkt, `awsexamplebucket1`, nur vom VPC-Endpunkt mit der ID `vpce-1a2b3c4d`. Die Richtlinie lehnt sämtlichen Zugriff auf den Bucket ab, der nicht über den angegebenen Endpunkt erfolgt. Der Endpunkt wird über die `aws:SourceVpce`-Bedingung festgelegt. Die `aws:SourceVpce`-Bedingung erfordert keinen Amazon-Ressourcennamen (ARN) für die VPC-Endpunkt-Ressource, sondern nur die VPC-Endpunkt-ID. Weitere Informationen über die Verwendung von Bedingungen in einer Richtlinie finden Sie unter Beispiele für Amazon-S3-Bedingungsschlüssel.

> **Important**
>
> - Bevor Sie die folgende Beispielrichtlinie verwenden, ersetzen Sie die VPC-Endpunkt-ID durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls können Sie nicht auf Ihren Bucket zugreifen.
> - Diese Richtlinie deaktiviert den Konsolenzugriff auf den angegebenen Bucket, da Konsolenanforderungen nicht vom angegebenen VPC-Endpunkt stammen.

```
{
   "Version": "2012-10-17",
   "Id": "Policy1415115909152",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPCE-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
       "Resource": ["arn:aws:s3:::awsexamplebucket1",
                    "arn:aws:s3:::awsexamplebucket1/*"],
       "Condition": {
         "StringNotEquals": {
           "aws:SourceVpce": "vpce-1a2b3c4d"
         }
       }
     }
   ]
}
```

Beschränkung des Zugriffs auf eine bestimmte VPC

Sie können eine Bucket-Richtlinie mit der `aws:SourceVpc`-Bedingung erstellen, die den Zugriff auf eine bestimmte VPC beschränkt. Dies ist hilfreich, wenn Sie mehrere VPC-Endpunkte innerhalb derselben VPC konfiguriert haben und den Zugriff auf Amazon-S3-Buckets für alle Endpunkte verwalten möchten. Nachfolgend finden Sie eine Beispielrichtlinie, die `awsexamplebucket1` den Zugriff auf seine Objekte von Benutzern außerhalb der VPC `vpc-111bbb22` verweigert. Die Richtlinie lehnt sämtlichen Zugriff auf den Bucket ab, der nicht über die angegebene VPC erfolgt. Diese Anweisung gewährt keinen Zugriff, dafür müssen Sie eine separate Allow-Anweisung

hinzufügen. Für den `vpc-111bbb22`-Bedingungsschlüssel wird kein ARN für die VPC-Ressource benötigt, sondern nur die VPC-ID.

> **Important**
>
> - Bevor Sie die folgende Beispielrichtlinie verwenden, ersetzen Sie die VPC-ID durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls können Sie nicht auf Ihren Bucket zugreifen.
> - Diese Richtlinie deaktiviert den Konsolenzugriff auf den angegebenen Bucket, da Konsolenanforderungen nicht von der angegebenen VPC stammen.

```
{
   "Version": "2012-10-17",
   "Id": "Policy1415115909153",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPC-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
       "Resource": ["arn:aws:s3:::awsexamplebucket1",
                    "arn:aws:s3:::awsexamplebucket1/*"],
       "Condition": {
         "StringNotEquals": {
           "aws:SourceVpc": "vpc-111bbb22"
         }
       }
     }
   ]
}
```

Beispiele für Bucket-Richtlinien

Mit Amazon-S3-Bucket-Richtlinien können Sie den Zugriff auf Objekte in Ihren Buckets sichern, sodass nur Benutzer mit den entsprechenden Berechtigungen darauf zugreifen können. Sie können sogar verhindern, dass authentifizierte Benutzer ohne die entsprechenden Berechtigungen auf Ihre Amazon-S3-Ressourcen zugreifen.

Dieser Abschnitt veranschaulicht Beispiele für typische Anwendungsfälle für Bucket-Richtlinien. Diese Beispielrichtlinien verwenden *DOC-EXAMPLE-BUCKET* als Ressourcenwert. Wenn Sie diese Richtlinien testen möchten, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen (z. B. Ihren Bucket-Namen).

Wenn Sie einer Gruppe von Objekten Berechtigungen erteilen oder verweigern möchten, können Sie Platzhalterzeichen für (*) Amazon-Ressourcennamen (ARNs) und andere Werte verwenden. Sie können beispielsweise den Zugriff auf Gruppen von Objekten steuern, die mit einem gemeinsamen Präfixbeginnen oder mit einer bestimmten Erweiterung wie `.html` enden.

Weitere Informationen zur AWS Identity and Access Management (IAM)-Richtliniensprache finden Sie unter Richtlinien und Berechtigungen in Amazon S3.

> **Note**
>
> Beim Testen von Berechtigungen unter Verwendung der Amazon-S3-Konsole erteilen Sie zusätzliche Berechtigungen, die die Konsole benötigt – `s3:ListAllMyBuckets`, `s3:GetBucketLocation` und `s3:ListBucket`. Ein detailliertes Beispiel für eine Richtlinie, die Berechtigungen für Benutzer erteilt und diese Berechtigungen unter Verwendung der Konsole testet, finden Sie unter Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien.

Zusätzliche Ressourcen für die Erstellung von Bucket-Richtlinien

- Eine Liste der IAM-Richtlinienaktionen, Ressourcen und Bedingungsschlüssel, die Sie beim Erstellen einer Bucket-Richtlinie verwenden können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.
- Anleitungen zur Erstellung einer S3-Richtlinie finden Sie unter Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole.
- Informationen zur Behebung von Fehlern mit einer Richtlinie finden Sie unter Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3.

Themen

- Verwaltung des Objektzugriffs mit Objektmarkierung
- Verwalten des Objektzugriffs mithilfe globaler Bedingungsschlüssel
- Verwalten des Zugriffs auf der Grundlage bestimmter IP-Adressen
- Verwalten des Zugriffs auf der Grundlage von HTTP- oder HTTPS-Anforderungen
- Verwalten des Benutzerzugriffs auf bestimmte Ordner
- Verwalten des Zugriffs für Zugriffsprotokolle
- Verwalten des Zugriffs auf eine Amazon CloudFront OAI
- Verwalten des Zugriffs für Amazon S3 Storage Lens
- Verwalten von Berechtigungen für S3 Inventory, S3 Analytics und S3-Inventory-Berichte
- Verlangen von MFA

Erteilen von Leseberechtigungen an einen anonymen Benutzer

Sie können Ihre Richtlinieneinstellungen verwenden, um öffentlichen anonymen Benutzern Zugriff zu gewähren. Dies ist nützlich, wenn Sie Ihren Bucket als statische Website konfigurieren. Dazu müssen Sie die Option Öffentlichen Zugriff blockieren für Ihren Bucket deaktivieren. Weitere Information dazu sowie zur erforderlichen Richtlinie finden Sie unter Festlegen von Berechtigungen für den Website-Zugriff. Informationen zum Einrichten restriktiverer Richtlinien für denselben Zweck finden Sie unter Wie kann ich nur für einige Objekte in meinem Amazon-S3-Bucket öffentlichen Lesezugriff gewähren?.

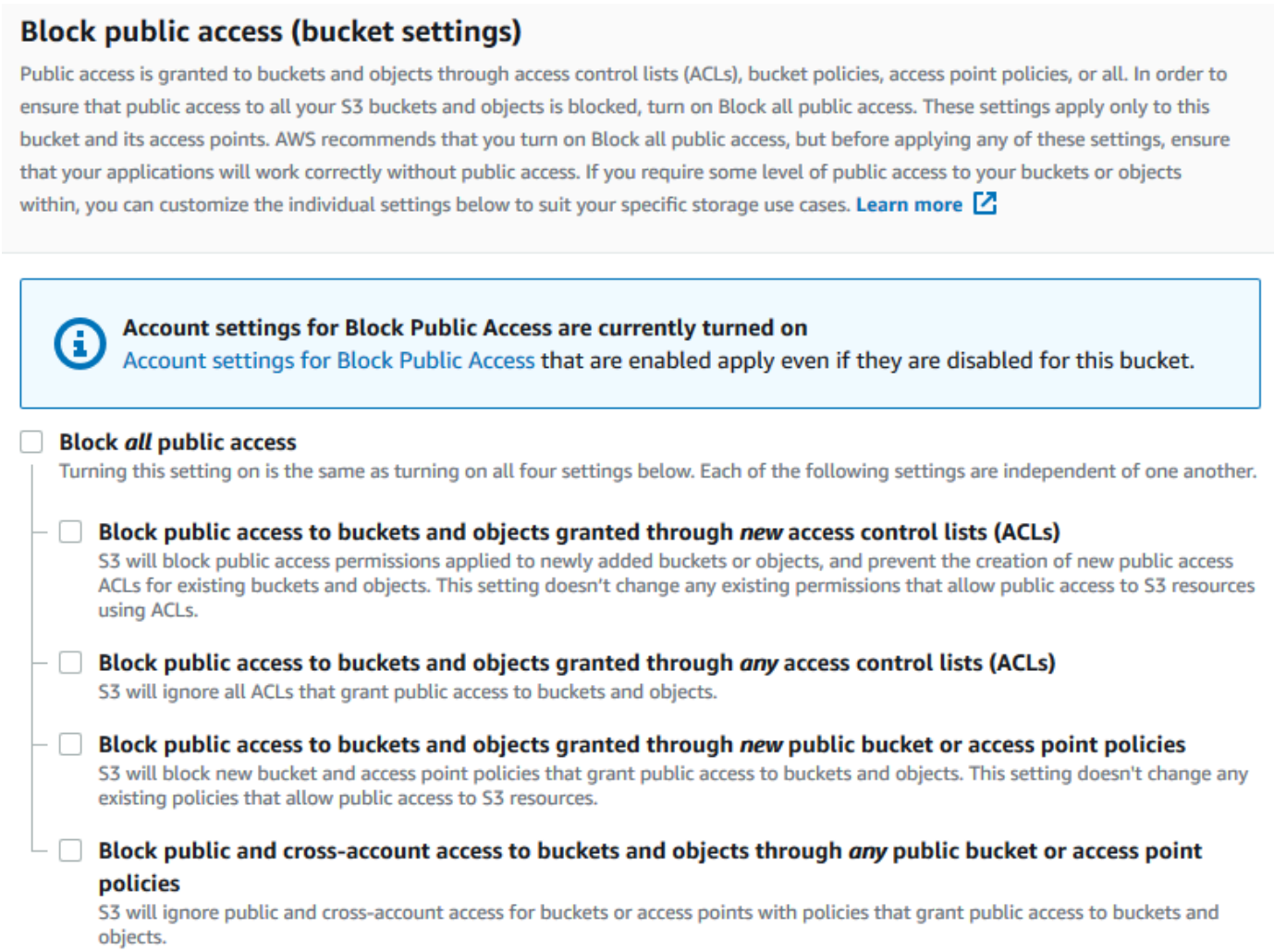
Standardmäßig blockiert Amazon S3 den öffentlichen Zugriff auf Ihr Konto und Ihre Buckets. Wenn Sie einen Bucket verwenden möchten, um eine statische Website zu hosten, können Sie diese Schritte verwenden, um Ihre Einstellungen für Block Public Access zu bearbeiten:

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Permissions (Berechtigungen).
4. Wählen Sie unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)), die Option Edit (Bearbeiten).
5. Löschen Sie Block all public access (Gesamten öffentlichen Zugriff blockieren) und wählen Sie Save (Speichern).

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.



Amazon S3 deaktiviert die Block Public Access-Einstellungen für Ihren Bucket. Um eine öffentliche, statische Website zu erstellen, müssen Sie möglicherweise auch die Block Public Access-Einstellungen für Ihr Konto bearbeiten, bevor Sie eine Bucket-Richtlinie hinzufügen. Wenn Kontoeinstellungen für Block Public Access derzeit aktiviert sind, wird unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)) ein Hinweis angezeigt.

Erfordern von Verschlüsselung

SSE-KMS für alle in einen Bucket geschriebenen Objekte verlangen

Die folgende Beispielrichtlinie erfordert, dass jedes Objekt, das in den Bucket geschrieben wird, mit serverseitiger Verschlüsselung unter Verwendung von AWS Key Management Service (AWS KMS)-

Schlüsseln (SSE-KMS) verschlüsselt wird. Wenn das Objekt nicht mit SSE-KMS verschlüsselt ist, wird die Anforderung abgelehnt.

```
{
"Version": "2012-10-17",
"Id": "PutObjPolicy",
"Statement": [{
  "Sid": "DenyObjectsThatAreNotSSEKMS",
  "Principal": "*",
  "Effect": "Deny",
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
  "Condition": {
    "Null": {
      "s3:x-amz-server-side-encryption-aws-kms-key-id": "true"
    }
  }
}]
}
```

Erfordern von SSE-KMS mit einem bestimmten AWS KMS key für alle in einen Bucket geschriebenen Objekte

Die folgende Beispielrichtlinie verhindert, dass Objekte in den Bucket geschrieben werden, wenn sie nicht mit SSE-KMS unter Verwendung einer bestimmten KMS-Schlüssel-ID verschlüsselt sind. Selbst wenn die Objekte mit SSE-KMS verschlüsselt sind, indem ein Pro-Anfrage-Header oder eine Standardverschlüsselung für Buckets verwendet wird, können die Objekte nicht in den Bucket geschrieben werden, wenn sie nicht mit dem angegebenen KMS-Schlüssel verschlüsselt wurden. Stellen Sie sicher, dass Sie den in diesem Beispiel verwendeten KMS-Schlüssel-ARN durch Ihren eigenen KMS-Schlüssel-ARN ersetzen.

```
{
"Version": "2012-10-17",
"Id": "PutObjPolicy",
"Statement": [{
  "Sid": "DenyObjectsThatAreNotSSEKMSWithSpecificKey",
  "Principal": "*",
  "Effect": "Deny",
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
  "Condition": {
    "ArnNotEqualsIfExists": {
```

```
      "s3:x-amz-server-side-encryption-aws-kms-key-id": "arn:aws:kms:us-
east-2:111122223333:key/01234567-89ab-cdef-0123-456789abcdef"
    }
  }
}]
}
```

Verwalten von Buckets mithilfe vordefinierter ACLs

Erteilung von Berechtigungen für mehrere Konten zum Hochladen von Objekten oder zum Festlegen von Objekt-ACLs für den öffentlichen Zugriff

Die folgende Beispielrichtlinie gewährt die `s3:PutObjectAcl` Berechtigungen `s3:PutObject` und für mehrere AWS-Konten und erfordert, dass alle Anforderungen für diese Operationen die `public-read` vordefinierte Zugriffskontrollliste (ACL) enthalten müssen. Weitere Informationen finden Sie unter Amazon S3-Richtlinienaktionen und Beispiele für Amazon-S3-Bedingungsschlüssel.

> **Warning**
>
> Mit der vordefinierten `public-read`-ACL kann jeder Benutzer weltweit die Objekte in Ihrem Bucket sehen. Seien Sie vorsichtig, wenn Sie anonymen Zugriff auf Ihren Amazon-S3-Bucket gewähren oder die Block-Einstellungen für den öffentlichen Zugriff deaktivieren. Wenn Sie anonymen Zugriff gewähren, kann jeder auf der ganzen Welt auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, niemals anonymen Zugriff auf Ihren Amazon-S3-Bucket zu gewähren, es sei denn, Sie müssen dies ausdrücklich tun, z. B. beim Hosting von statischen Websites. Informationen zum Aktivieren der Einstellungen zum Blockieren des öffentlichen Zugriffs für das Hosten statischer Websites finden Sie unter Tutorial: Konfiguration einer statischen Website in Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AddPublicReadCannedAcl",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root",
                    "arn:aws:iam::444455556666:root"
```

```
                ]
            },
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ]
                }
            }
        }
    ]
}
```

Erteilung von kontoübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt

Das folgende Beispiel zeigt, wie Sie einem anderen das Hochladen AWS-Konto von Objekten in Ihren Bucket ermöglichen und gleichzeitig sicherstellen, dass Sie die volle Kontrolle über die hochgeladenen Objekte haben. Diese Richtlinie gewährt einem bestimmten AWS-Konto (*111122223333*) nur dann die Möglichkeit, Objekte hochzuladen, wenn dieses Konto beim Hochladen die `bucket-owner-full-control` vordefinierte ACL enthält. Die `StringEquals`-Bedingung in der Richtlinie spezifiziert den `s3:x-amz-acl`-Bedingungsschlüssel, um die vordefinierte ACL-Anforderung auszudrücken. Weitere Informationen finden Sie unter [Beispiele für Amazon-S3-Bedingungsschlüssel].

```
{
   "Version":"2012-10-17",
   "Statement":[
     {
       "Sid":"PolicyForAllowUploadWithACL",
       "Effect":"Allow",
       "Principal":{"AWS":"111122223333"},
       "Action":"s3:PutObject",
       "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
       "Condition": {
         "StringEquals": {"s3:x-amz-acl":"bucket-owner-full-control"}
       }
```

```
      }
   ]
}
```

Verwaltung des Objektzugriffs mit Objektmarkierung

Einem Benutzer nur das Lesen von Objekten gestatten, die einen bestimmten Tag-Schlüssel und -Wert besitzen

Die folgende Berechtigungsrichtlinie beschränkt einen Benutzer darauf, nur Objekte zu lesen, die den Tag-Schlüssel und -Wert `environment: production` haben. Diese Richtlinie verwendet den `s3:ExistingObjectTag`-Bedingungsschlüssel, um den Tag-Schlüssel und -Wert anzugeben.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Principal":{
            "AWS":"arn:aws:iam::111122223333:role/JohnDoe"
         },
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
         "Condition":{
            "StringEquals":{
               "s3:ExistingObjectTag/environment":"production"
            }
         }
      }
   ]
}
```

Einschränken, welche Objekt-Tag-Schlüssel Benutzer hinzufügen können

Die folgende Beispielrichtlinie erteilt einem Benutzer die Berechtigungen, die `s3:PutObjectTagging`-Aktion auszuführen, die dem Benutzer gestattet, einem vorhandenen Objekt Markierungen hinzuzufügen. Die Bedingung verwendet den Bedingungsschlüssel `s3:RequestObjectTagKeys`, um die zulässigen Tag-Schlüssel zu spezifizieren, z. B. `Owner` oder

`CreationDate`. Weitere Informationen finden Sie unter Erstellen einer Bedingung, die mehrere Schlüsselwerte testet im IAM-Benutzerhandbuch.

Die Richtlinie stellt sicher, dass jeder in der Anfrage angegebene Tag-Schlüssel ein autorisierter Tag-Schlüssel ist. Der `ForAnyValue`-Qualifizierer in der Bedingung stellt sicher, dass mindestens einer der spezifizierten Schlüssel in der Anfrage enthalten ist.

```
{
   "Version": "2012-10-17",
  "Statement": [
    {"Principal":{"AWS":[
            "arn:aws:iam::111122223333:role/JohnDoe"
         ]
       },
 "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"ForAnyValue:StringEquals": {"s3:RequestObjectTagKeys": [
            "Owner",
            "CreationDate"
          ]
        }
      }
    }
  ]
}
```

Einen spezifischen Tag-Schlüssel und -Wert verlangen, wenn Benutzern erlaubt wird, Objekt-Tags hinzuzufügen

Die folgende Beispielrichtlinie erteilt einem Benutzer die Berechtigungen, die `s3:PutObjectTagging`-Aktion auszuführen, die dem Benutzer gestattet, einem vorhandenen Objekt Markierungen hinzuzufügen. Die Bedingung verlangt, dass der Benutzer einen spezifischen Tag-Schlüssel (z. B. *`Project`*) mit dem Wert *`X`* angibt.

```
{
   "Version": "2012-10-17",
  "Statement": [
```

```
    {"Principal":{"AWS":[
       "arn:aws:iam::111122223333:user/JohnDoe"
         ]
       },
      "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"StringEquals": {"s3:RequestObjectTag/Project": "X"
        }
      }
    }
  ]
}
```

Einem Benutzer nur das Hinzufügen mit einem bestimmten Tag-Schlüssel und -Wert erlauben

Die folgende Beispielrichtlinie gewährt einem Benutzer die Berechtigung, die `s3:PutObject`-Aktion auszuführen, sodass er Objekte einem Bucket hinzufügen kann. Die `Condition`-Anweisung schränkt die Tag-Schlüssel und -Werte jedoch ein, die für die hochgeladenen Objekte zulässig sind. In diesem Beispiel kann der Benutzer dem Bucket nur Objekte hinzufügen, die den spezifischen Tag-Schlüssel (*`Department`*) mit dem Wert *`Finance`* haben.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Principal":{
            "AWS":[
                "arn:aws:iam::111122223333:user/JohnDoe"
         ]
        },
        "Effect": "Allow",
        "Action": [
            "s3:PutObject"
        ],
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        ],
        "Condition": {
            "StringEquals": {
```

```
                "s3:RequestObjectTag/Department": "Finance"
            }
        }
    }]
}
```

Verwalten des Objektzugriffs mithilfe globaler Bedingungsschlüssel

Globale Bedingungsschlüssel sind Bedingungskontextschlüssel mit einem `aws` Präfix. AWS-Services kann globale Bedingungsschlüssel oder servicespezifische Schlüssel unterstützen, die das Servicepräfix enthalten. Sie können das `Condition`-Element einer JSON-Richtlinie verwenden, um Schlüssel in einer Anforderung mit Schlüsselwerten zu vergleichen, die Sie in Ihrer Richtlinie angeben.

Beschränken des Zugriffs nur auf Zugriffsprotokollbereitstellungen von Amazon S3 Server

In der folgenden Beispiel-Bucket-Richtlinie wird der aws:SourceArn globale Bedingungsschlüssel verwendet, um den Amazon-Ressourcennamen (ARN) der Ressource zu vergleichen und eine service-to-service Anforderung mit dem in der Richtlinie angegebenen ARN zu stellen. Sie können diesen globalen `aws:SourceArn`-Bedingungsschlüssel verwenden, um zu verhindern, dass der Amazon-S3-Service bei Transaktionen zwischen Services als verwechselter Stellvertreter verwendet wird. Nur der Amazon-S3-Service darf dem Amazon-S3-Bucket Objekte hinzufügen.

Diese Bucket-Beispielrichtlinie gewährt nur dem Prinzipal des Protokollierungsservices (`logging.s3.amazonaws.com`) `s3:PutObject`-Berechtigungen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutObjectS3ServerAccessLogsPolicy",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET-logs/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111111111111"
                },
```

```
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::EXAMPLE-SOURCE-BUCKET"
                }
            }
        },
        {
            "Sid": "RestrictToS3ServerAccessLogs",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET-logs/*",
            "Condition": {
                "ForAllValues:StringNotEquals": {
                    "aws:PrincipalServiceNamesList": "logging.s3.amazonaws.com"
                }
            }
        }
    ]
}
```

Erlauben des Zugriffs nur für Ihre Organisation

Wenn Sie verlangen möchten, dass alle IAM-Prinzipale, die auf eine Ressource zugreifen, von einem AWS-Konto in Ihrer Organisation (einschließlich des AWS Organizations Verwaltungskontos) stammen, können Sie den `aws:PrincipalOrgID` globalen Bedingungsschlüssel verwenden.

Wenn Sie diese Art von Zugriff gewähren oder einschränken möchten, definieren Sie die `aws:PrincipalOrgID`-Bedingung und legen Sie den Wert in der Bucket-Richtlinie auf Ihre Organisations-ID fest. Die Organisations-ID wird verwendet, um den Zugriff auf den Bucket zu kontrollieren. Wenn Sie die `aws:PrincipalOrgID`-Bedingung verwenden, werden die Berechtigungen aus der Bucket-Richtlinie auch auf alle neuen Konten angewendet, die der Organisation hinzugefügt werden.

Hier ist ein Beispiel für eine auf Ressourcen basierende Bucket-Richtlinie, die Sie verwenden können, um bestimmten IAM-Prinzipalen in Ihrer Organisation direkten Zugriff auf Ihren Bucket zu erteilen. Durch das Hinzufügen des globalen Bedingungsschlüssels `aws:PrincipalOrgID` zu Ihrer Bucket-Richtlinie muss sich das Prinzipal-Konto nun in Ihrer Organisation befinden, um Zugriff auf die Ressource zu erhalten. Selbst wenn Sie bei der Gewährung des Zugriffs versehentlich ein falsches Konto angeben, fungiert der globale Bedingungsschlüssel aws:PrincipalOrgID als zusätzliche Absicherung. Wenn dieser globale Schüssel in einer Richtlinie verwendet wird, verhindert diese, dass alle Prinzipale außerhalb der angegebenen Organisation auf den Amazon-S3-Bucket

zugreifen können. Nur Prinzipale von Konten in der aufgelisteten Organisation erhalten Zugriff auf die Ressource.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AllowGetObject",
        "Principal": {
            "AWS": "*"
        },
        "Effect": "Allow",
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalOrgID": ["o-aa111bb222"]
            }
        }
    }]
}
```

Verwalten des Zugriffs auf der Grundlage bestimmter IP-Adressen

Beschränken des Zugriffs auf bestimmte IP-Adressen

Im folgenden Beispiel werden allen Benutzern die Berechtigung zum Ausführen von Amazon-S3-Vorgängen an Objekten in festgelegten Buckets verweigert, es sei denn, die Anforderung stammt aus dem in der Bedingung angegebenen IP-Adressbereich.

> **Note**
>
> Wenn Sie den Zugriff auf eine bestimmte IP-Adresse beschränken, stellen Sie sicher, dass Sie auch angeben, welche VPC-Endpunkte, VPC-Quell-IP-Adressen oder externen IP-Adressen auf den S3-Bucket zugreifen können. Andernfalls verlieren Sie möglicherweise den Zugriff auf den Bucket, wenn Ihre Richtlinie allen Benutzern die Ausführung von S3-Operationen an Objekten im Bucket verweigert, ohne dass bereits die entsprechenden Berechtigungen vorhanden sind.

Die `Condition`-Anweisung dieser Richtlinie identifiziert *`192.0.2.0/24`* als den Bereich zulässiger IP-Adressen des Internetprotokolls Version 4 (IPv4).

Der -ConditionBlock verwendet die -NotIpAddressBedingung und den -aws:SourceIpBedingungsschlüssel, bei dem es sich um einen AWS weiten Bedingungsschlüssel handelt. Der aws:SourceIp-Bedingungsschlüssel kann nur für öffentliche IP-Adressbereiche verwendet werden. Weitere Informationen über diese Bedingungsschlüssel finden Sie unter Beispiele für Amazon-S3-Bedingungsschlüssel. Die aws:SourceIp-IPv4-Werte verwenden die CIDR-Standardnotation. Weitere Informationen finden Sie in der IAM-JSON-Richtlinienelemente-Referenz im IAM-Benutzerhandbuch.

> **Warning**
>
> Ersetzen Sie vor der Verwendung dieser Richtlinie den *192.0.2.0/24*-IP-Adressbereich in diesem Beispiel durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls verlieren Sie die Möglichkeit, auf Ihren Bucket zuzugreifen.

```
{
    "Version": "2012-10-17",
    "Id": "S3PolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                }
            }
        }
    ]
}
```

Zulassen von IPv4- und IPv6-Adressen

Wenn Sie mit der Verwendung von IPv6-Adressen beginnen, empfehlen wir, dass Sie alle Richtlinien Ihrer Organisation zusätzlich zu Ihren bereits vorhandenen IPv4-Adressbereichen auf Ihre IPv6-

Adressbereiche aktualisieren. Auf diese Weise können Sie sicherstellen, dass die Richtlinien auch während der Umstellung auf IPv6 weiterhin funktionieren.

Das folgende Beispiel für eine Bucket-Richtlinie zeigt, wie Sie IPv4- und IPv6-Adressbereiche kombinieren können, um alle gültigen IP-Adressen in Ihrer Organisation abzudecken. Die Beispielrichtlinie erteilt Zugriff auf die IP-Adressen *192.0.2.1* und *2001:DB8:1234:5678::1* und verweigert den Zugriff auf die Adressen *203.0.113.1* und *2001:DB8:1234:5678:ABCD::1*.

Der `aws:SourceIp`-Bedingungsschlüssel kann nur für öffentliche IP-Adressbereiche verwendet werden. Die IPv6-Werte für `aws:SourceIp` müssen im CIDR-Standardformat angegeben werden. Für IPv6 unterstützen wir die Verwendung von `::` zur Darstellung eines Bereichs von Nullen (z. B. `2001:DB8:1234:5678::/64`). Weitere Informationen finden Sie unter IP-Adressen-Bedingungsoperatoren im IAM-Benutzerhandbuch.

> **Warning**
>
> Ersetzen Sie die IP-Adressbereiche in diesem Beispiel durch geeignete Werte für Ihren Anwendungsfall, bevor Sie diese Richtlinie verwenden. Andernfalls verlieren Sie möglicherweise die Möglichkeit, auf Ihren Bucket zuzugreifen.

```
{
    "Id": "PolicyId2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIPmix",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                },
```

```
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678:ABCD::/80"
                    ]
                }
            }
        }
    ]
}
```

Verwalten des Zugriffs auf der Grundlage von HTTP- oder HTTPS-Anforderungen

Beschränken des Zugriffs nur auf HTTPS-Anforderungen

Wenn Sie verhindern möchten, dass potenzielle Angreifer den Netzwerkverkehr manipulieren, können Sie HTTPS (TLS) verwenden, um nur verschlüsselte Verbindungen zuzulassen und gleichzeitig den Zugriff von HTTP-Anforderungen auf Ihren Bucket zu beschränken. Um festzustellen, ob es sich bei der Anforderung um HTTP oder HTTPS handelt, verwenden Sie den globalen Bedingungsschlüssel aws:SecureTransport in Ihrer S3-Bucket-Richtlinie. Der `aws:SecureTransport`-Bedingungsschlüssel überprüft, ob eine Anforderung über HTTP gesendet wurde.

Wenn eine Anforderung `true` zurückgibt, wurde sie über HTTPS gesendet. Wenn eine Anforderung `false` zurückgibt, wurde sie über HTTP gesendet. Sie können dann den Zugriff auf Ihren Bucket basierend auf dem gewünschten Anforderungsschema erlauben oder verweigern.

Im folgenden Beispiel verweigert die Bucket-Richtlinie ausdrücklich HTTP-Anforderungen.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "RestrictToTLSRequestsOnly",
        "Action": "s3:*",
        "Effect": "Deny",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        ],
        "Condition": {
            "Bool": {
                "aws:SecureTransport": "false"
```

```
            }
        },
        "Principal": "*"
    }]
}
```

Beschränken des Zugriffs auf einen spezifischen HTTP-Referer

Angenommen, Sie haben eine Website mit dem Domänennamen *www.example.com* oder *example.com* mit Links zu Fotos und Videos, die in Ihrem Bucket namens *DOC-EXAMPLE-BUCKET* gespeichert sind. Standardmäßig sind alle Amazon S3-Ressourcen privat, sodass nur das , das die Ressourcen erstellt hat AWS-Konto , darauf zugreifen kann.

Wenn Sie Lesezugriff auf diese Objekte von Ihrer Website aus erlauben möchten, können Sie eine Bucket-Richtlinie hinzufügen, die die `s3:GetObject`-Berechtigung mit der Bedingung erlaubt, dass die `GET`-Anforderung von bestimmten Webseiten stammen muss. Die folgende Richtlinie schränkt Anforderungen ein, indem die Bedingung `StringLike` zusammen mit dem `aws:Referer`-Bedingungsschlüssel verwendet wird.

```
{
  "Version":"2012-10-17",
  "Id":"HTTP referer policy example",
  "Statement":[
    {
      "Sid":"Allow only GET requests originating from www.example.com and
 example.com.",
      "Effect":"Allow",
      "Principal":"*",
      "Action":["s3:GetObject","s3:GetObjectVersion"],
      "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
      "Condition":{
        "StringLike":{"aws:Referer":["http://www.example.com/*","http://example.com/
*"]}
      }
    }
  ]
}
```

Stellen Sie sicher, dass die von Ihnen verwendeten Browser den HTTP-`referer`-Header in der Anforderung enthalten.

> **Warning**
>
> Wir empfehlen, bei der Verwendung des `aws:Referer`-Bedingungsschlüssels vorsichtig zu sein. Ein öffentlich bekannter HTTP-Referer-Header-Wert sollte möglichst nicht eingeschlossen werden. Nicht autorisierte Parteien können mit modifizierten oder benutzerdefinierten Browsern einen beliebigen `aws:Referer`-Wert ihrer Wahl bereitstellen. Verwenden Sie daher nicht , `aws:Referer` um zu verhindern, dass Unbefugte direkte AWS Anfragen stellen.
>
> Der `aws:Referer`-Bedingungsschlüssel wird nur bereitgestellt, damit Kunden ihre digitalen, in Amazon S3 gespeicherten Inhalte vor der Referenzierung auf nicht autorisierte Drittanbieter-Websites schützen können. Weitere Informationen finden Sie unter aws:Referer im IAM-Benutzerhandbuch.

Verwalten des Benutzerzugriffs auf bestimmte Ordner

Erteilen des Benutzerzugriffs auf bestimmte Ordner

Angenommen, Sie versuchen, Benutzern Zugriff auf einen bestimmten Ordner zu gewähren. Wenn der IAM-Benutzer und der S3-Bucket zum selben gehören AWS-Konto, können Sie eine IAM-Richtlinie verwenden, um dem Benutzer Zugriff auf einen bestimmten Bucket-Ordner zu gewähren. Bei diesem Ansatz müssen Sie Ihre Bucket-Richtlinie nicht aktualisieren, um Zugriff zu gewähren. Sie können die IAM-Richtlinie einer IAM-Rolle hinzufügen, zu der mehrere Benutzer wechseln können.

Wenn die IAM-Identität und der S3-Bucket zu verschiedenen gehören AWS-Konten, müssen Sie sowohl in der IAM-Richtlinie als auch in der Bucket-Richtlinie kontoübergreifenden Zugriff gewähren. Weitere Informationen zum Gewähren von kontoübergreifendem Zugriff finden Sie unter Bucket-Eigentümer erteilt kontoübergreifende Bucket-Berechtigungen.

Die folgende Bucket-Beispielrichtlinie gewährt *JohnDoe* vollen Konsolenzugriff nur auf seinen Ordner (`home/`*JohnDoe*`/`). Indem Sie einen `home`-Ordner erstellen und Ihren Benutzern die entsprechenden Berechtigungen gewähren, können sich mehrere Benutzer einen einzelnen Bucket teilen. Diese Richtlinie besteht aus drei `Allow`-Anweisungen:

- *AllowRootAndHomeListingOfCompanyBucket*: Erlaubt dem Benutzer (*JohnDoe*), Objekte auf der Stammebene des *DOC-EXAMPLE-BUCKET*-Buckets und im `home`-Ordner aufzulisten. Diese Anweisung erlaubt dem Benutzer außerdem, mithilfe der Konsole nach dem Präfix `home/` zu suchen.

- *AllowListingOfUserFolder*: Erlaubt dem Benutzer (*JohnDoe*), alle Objekte im home/*JohnDoe*/-Ordner und entsprechenden Unterordnern aufzulisten.
- *AllowAllS3ActionsInUserFolder*: Erlaubt dem Benutzer, alle Amazon-S3-Aktionen durchzuführen, indem `Read`-, `Write`- und `Delete`-Berechtigungen gewährt werden. Die Berechtigungen sind auf den Home-Ordner des Bucket-Eigentümers beschränkt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRootAndHomeListingOfCompanyBucket",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/JohnDoe"
                ]
            },
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
            "Condition": {
                "StringEquals": {
                    "s3:prefix": ["", "home/", "home/JohnDoe"],
                    "s3:delimiter": ["/"]
                }
            }
        },
        {
            "Sid": "AllowListingOfUserFolder",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/JohnDoe"
                ]
            },
            "Action": ["s3:ListBucket"],
            "Effect": "Allow",
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
            "Condition": {
                "StringLike": {
                    "s3:prefix": ["home/JohnDoe/*"]
                }
            }
```

```
        },
        {
            "Sid": "AllowAllS3ActionsInUserFolder",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/JohnDoe"
                ]
            },
            "Action": ["s3:*"],
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/home/JohnDoe/*"]
        }
    ]
}
```

Verwalten des Zugriffs für Zugriffsprotokolle

Gewähren des Zugriffs auf Application Load Balancer, um Zugriffsprotokolle zu aktivieren

Wenn Sie die Zugriffsprotokollierung für Ihren Application Load Balancer aktivieren, müssen Sie den Namen des S3-Buckets angeben, in dem der Load Balancer die Protokolle speichert. Dem Bucket muss eine Richtlinie angefügt sein, die Elastic Load Balancing die Berechtigung zum Schreiben in den Bucket gewährt.

Im folgenden Beispiel gewährt die Bucket-Richtlinie Elastic Load Balancing (ELB) die Berechtigung, die Zugriffsprotokolle in den Bucket zu schreiben:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": "arn:aws:iam::elb-account-id:root"
      },
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/AWSLogs/111122223333/*"
    }
  ]
}
```

> **Note**
>
> Stellen Sie sicher, dass Sie *elb-account-id* durch die AWS-Konto -ID für Elastic Load Balancing für Ihre AWS-Region ersetzen. Eine Liste der Regionen von Elastic Load Balancing finden Sie unter Hinzufügen von Richtlinien zu Ihrem S3-Bucket im Benutzerhandbuch für Elastic Load Balancing.

Wenn Ihr AWS-Region nicht in der Liste der unterstützten Regionen von Elastic Load Balancing angezeigt wird, verwenden Sie die folgende Richtlinie, die dem angegebenen Protokollbereitstellungsservice Berechtigungen erteilt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
       "Principal": {
         "Service": "logdelivery.elasticloadbalancing.amazonaws.com"
          },
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/AWSLogs/111122223333/*"
    }
  ]
}
```

Stellen Sie anschließend sicher, dass Sie Ihre Zugriffsprotokolle von Elastic Load Balancing konfigurieren, indem Sie sie aktivieren. Sie können Ihre Bucket-Berechtigungen überprüfen, indem Sie eine Testdatei erstellen.

Verwalten des Zugriffs auf eine Amazon CloudFront OAI

Erteilen der Berechtigung für eine Amazon CloudFront OAI

Die folgende Beispiel-Bucket-Richtlinie erteilt einer CloudFront Ursprungszugriffsidentität (OAI) die Berechtigung, alle Objekte in Ihrem S3-Bucket abzurufen (zu lesen). Sie können eine CloudFront OAI verwenden, um Benutzern den Zugriff auf Objekte in Ihrem Bucket über , CloudFront aber nicht direkt über Amazon S3 zu ermöglichen. Weitere Informationen finden Sie unter Beschränken des Zugriffs auf Amazon S3-Inhalte mithilfe einer Ursprungszugriffsidentität im Amazon- CloudFront Entwicklerhandbuch.

Bei der folgenden Richtlinie wird die ID der OAI als `Principal` der Richtlinie verwendet. Weitere Informationen zur Verwendung von S3-Bucket-Richtlinien zum Gewähren des Zugriffs auf eine CloudFront OAI finden Sie unter Migrieren von der Ursprungszugriffsidentität (OAI) zur Ursprungszugriffssteuerung (OAC) im Amazon- CloudFront Entwicklerhandbuch.

Zur Verwendung dieses Beispiels gehen Sie wie folgt vor:

- Ersetzen Sie *EH1HDMB1FH2TC* mit der OAI-ID. Die ID der OAI finden Sie auf der Seite Origin Access Identity in der - CloudFront Konsole oder verwenden Sie ListCloudFrontOriginAccessIdentities in der CloudFront API.
- Ersetzen Sie *DOC-EXAMPLE-BUCKET* durch den Namen von Ihrem Bucket.

```
{
    "Version": "2012-10-17",
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity EH1HDMB1FH2TC"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Verwalten des Zugriffs für Amazon S3 Storage Lens

Erteilen von Berechtigungen für Amazon S3 Storage Lens

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Objekt- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.

S3 Storage Lens kann Ihre aggregierten Speichernutzungsmetriken in einen Amazon-S3-Bucket exportieren, um sie weiter zu analysieren. Der Bucket, in dem S3 Storage Lens seine Metrikexporte speichert, wird als Ziel-Bucket bezeichnet. Sie müssen eine Bucket-Richtlinie für den Ziel-Bucket haben, wenn Sie Ihren Metrikexport in S3 Storage Lens einrichten. Weitere Informationen finden Sie unter Bewerten Ihrer Speicheraktivität und -nutzung mit Amazon S3 Storage Lens.

Die folgende Bucket-Beispielrichtlinie erteilt Amazon S3 die Berechtigung, Objekte (PUT-Anforderungen) in einen Ziel-Bucket zu schreiben. Sie verwenden eine solche Bucket-Richtlinie für den Ziel-Bucket, wenn Sie einen S3-Storage-Lens-Metrik-Export einrichten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3StorageLensExamplePolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "storage-lens.s3.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::destination-bucket/destination-prefix/
StorageLens/111122223333/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": "111122223333",
                    "aws:SourceArn": "arn:aws:s3:region-code:111122223333:storage-
lens/storage-lens-dashboard-configuration-id"
                }
            }
        }
    ]
}
```

Verwenden Sie die folgende Änderung an der vorherigen `Resource`-Anweisung der Bucket-Richtlinie beim Einrichten eines Metrikexports von S3 Storage Lens auf Organisationsebene.

```
"Resource": "arn:aws:s3:::destination-bucket/destination-prefix/StorageLens/your-
organization-id/*",
```

Verwalten von Berechtigungen für S3 Inventory, S3 Analytics und S3-Inventory-Berichte

Gewähren von Berechtigungen für S3 Inventory und S3 Analytics

S3 Inventory erstellt Listen der Objekte in einem Bucket und der Speicherklassenanalyse-Export von S3 Analytics erstellt Ausgabedateien der in der Analyse verwendeten Daten. Der Bucket, dessen Objekte die Bestandserfassung auflistet, wird als Quell-Bucket bezeichnet. Der Bucket, in dem die Bestandsdatei und die Analyseexportdatei geschrieben werden, wird als Ziel-Bucket bezeichnet. Sie müssen eine Bucket-Richtlinie für den Ziel-Bucket erstellen, wenn Sie den Bestand oder einen Analyseexport einrichten. Weitere Informationen finden Sie unter Amazon S3 Inventory und Amazon S3 analytics – Speicherklassen-Analyse.

Das folgende Beispiel für eine Bucket-Richtlinie erteilt Amazon S3 die Berechtigung, Objekte aus dem Konto für den Quell-Bucket in den Ziel-Bucket zu schreiben (PUT-Anforderungen). Sie verwenden eine solche Bucket-Richtlinie für den Ziel-Bucket, wenn Sie einen S3-Inventory- und S3-Analytics-Export einrichten.

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
            "Sid": "InventoryAndAnalyticsExamplePolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/*"
            ],
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
                },
                "StringEquals": {
                    "aws:SourceAccount": "111122223333",
                    "s3:x-amz-acl": "bucket-owner-full-control"
                }
            }
        }
```

```
    ]
}
```

Steuern der Erstellung von S3-Inventory-Berichten

Amazon S3 Inventory erstellt Listen der Objekte in einem S3-Bucket sowie die Metadaten für jedes Objekt. Die -`s3:PutInventoryConfiguration`Berechtigung ermöglicht es einem Benutzer, eine Bestandskonfiguration zu erstellen, die alle standardmäßig verfügbaren Objektmetadatenfelder enthält, und den Ziel-Bucket anzugeben, in dem der Bestand gespeichert werden soll. Ein Benutzer mit Lesezugriff auf Objekte im Ziel-Bucket kann auf alle Objektmetadatenfelder zugreifen, die im Bestandsbericht verfügbar sind. Weitere Informationen über die Metadatenfelder, die in S3 Inventory verfügbar sind, finden Sie unterAmazon-S3-Inventory-Liste.

Um einen Benutzer daran zu hindern, einen S3-Inventory-Bericht zu konfigurieren, entfernen Sie die -`s3:PutInventoryConfiguration`Berechtigung für den Benutzer.

Einige Objektmetadatenfelder in S3-Inventory-Berichtskonfigurationen sind optional, was bedeutet, dass sie standardmäßig verfügbar sind, aber eingeschränkt werden können, wenn Sie einem Benutzer die -`s3:PutInventoryConfiguration`Berechtigung erteilen. Mit dem `s3:InventoryAccessibleOptionalFields` Bedingungsschlüssel können Sie steuern, ob Benutzer diese optionalen Metadatenfelder in ihre Berichte aufnehmen können. Eine Liste der optionalen Metadatenfelder, die in S3 Inventory verfügbar sind, finden Sie unter OptionalFields in der API-Referenz zu Amazon Simple Storage Service.

Um einem Benutzer die Berechtigung zum Erstellen einer Bestandskonfiguration mit bestimmten optionalen Metadatenfeldern zu erteilen, verwenden Sie den `s3:InventoryAccessibleOptionalFields` Bedingungsschlüssel , um die Bedingungen in Ihrer Bucket-Richtlinie zu verfeinern.

Die folgende Beispielrichtlinie erteilt einem Benutzer (*Ana*) die Berechtigung, eine Bestandskonfiguration bedingt zu erstellen. Die -`ForAllValues:StringEquals`Bedingung in der Richtlinie verwendet den -`s3:InventoryAccessibleOptionalFields`Bedingungsschlüssel, um die beiden zulässigen optionalen Metadatenfelder anzugeben, `Size` und `StorageClass`. Wenn also eine Bestandskonfiguration *Ana* erstellt, können sie als einzige optionale Metadatenfelder `Size` und einschließen`StorageClass`.

```
{
 "Id": "InventoryConfigPolicy",
 "Version": "2012-10-17",
```

```
 "Statement": [{
   "Sid": "AllowInventoryCreationConditionally",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/Ana"
   },
   "Action":
    "s3:PutInventoryConfiguration",
   "Resource":
    "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
   "Condition": {
    "ForAllValues:StringEquals": {
     "s3:InventoryAccessibleOptionalFields": [
        "Size",
        "StorageClass"
        ]
      }
    }
   }
 ]
}
```

Um einen Benutzer daran zu hindern, einen S3-Inventory-Bericht zu konfigurieren, der bestimmte optionale Metadatenfelder enthält, fügen Sie der Bucket-Richtlinie für den Quell-Bucket eine explizite -DenyAnweisung hinzu. Die folgende Bucket*Ana*-Beispielrichtlinie verweigert dem Benutzer das Erstellen einer Bestandskonfiguration im Quell-Bucket*DOC-EXAMPLE-SOURCE-BUCKET*, die die optionalen `ObjectOwner` Metadatenfelder `ObjectAccessControlList` oder enthält. Der Benutzer *Ana* kann weiterhin eine Bestandskonfiguration mit anderen optionalen Metadatenfeldern erstellen.

```
{
 "Id": "InventoryConfigSomeFields",
 "Version": "2012-10-17",
 "Statement": [{
   "Sid": "AllowInventoryCreation",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/Ana"
   },
   "Action": "s3:PutInventoryConfiguration",
   "Resource":
    "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
```

```
  },
  {
   "Sid": "DenyCertainInventoryFieldCreation",
   "Effect": "Deny",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/Ana"
   },
   "Action": "s3:PutInventoryConfiguration",
   "Resource":
     "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
   "Condition": {
    "ForAnyValue:StringEquals": {
     "s3:InventoryAccessibleOptionalFields": [
        "ObjectOwner",
        "ObjectAccessControlList"
        ]
      }
    }
   }
 ]
}
```

Note

Die Verwendung des `s3:InventoryAccessibleOptionalFields` Bedingungsschlüssels in Bucket-Richtlinien hat keinen Einfluss auf die Bereitstellung von Bestandsberichten, die auf den vorhandenen Bestandskonfigurationen basieren.

Important

Es wird empfohlen, `ForAllValues` mit einem `-Allow`Effekt oder `ForAnyValue` mit einem `-Deny`Effekt zu verwenden, wie in den vorherigen Beispielen gezeigt.
Verwenden Sie nicht `ForAllValues` mit einem `-Deny`Effekt oder `ForAnyValue` mit einem `-Allow`Effekt, da diese Kombinationen zu restriktiv sein und das Löschen der Bestandskonfiguration blockieren können.
Weitere Informationen zu den `ForAnyValue` Bedingungssatz-Operatoren `ForAllValues` und finden Sie unter Mehrwertige Kontextschlüssel im IAM-Benutzerhandbuch.

Verlangen von MFA

Amazon S3 unterstützt MFA-geschützten API-Zugriff, eine Funktion, die eine Multi-Factor Authentication (MFA) für den Zugriff auf Ihre Amazon-S3-Ressourcen erzwingen kann. Die Multi-Faktor-Authentifizierung bietet ein zusätzliches Sicherheitsniveau, das Sie auf Ihre AWS Umgebung anwenden können. Die MFA handelt sich um eine Sicherheitsfunktion, die die Angabe eines gültigen MFA-Codes von Benutzern erfordert, mit dem das physische Eigentum eines MFA-Geräts belegt wird. Weitere Informationen finden Sie unter AWS Multi-Factor-Authentication. Sie können die MFA für alle Anfragen zum Zugriff auf Ihre Amazon-S3-Ressourcen verlangen.

Verwenden Sie zum Erzwingen der MFA-Anforderung den `aws:MultiFactorAuthAge`-Bedingungsschlüssel in einer Bucket-Richtlinie. IAM-Benutzer können auf Amazon S3-Ressourcen zugreifen, indem sie temporäre Anmeldeinformationen verwenden, die von der AWS Security Token Service () ausgestellt wurden AWS STS. Sie geben den MFA-Code zum Zeitpunkt der AWS STS -Anfrage an.

Wenn Amazon S3 eine Anforderung mit Multi-Factor Authentication empfängt, liefert der `aws:MultiFactorAuthAge`-Bedingungsschlüssel einen numerischen Wert, der angibt, wie lange (in Sekunden) es her ist, dass die temporären Anmeldeinformationen erstellt wurden. Wenn die in der Anforderung bereitgestellten temporären Anmeldeinformationen nicht mit einem MFA-Gerät erstellt wurden, ist dieser Schlüsselwert null (nicht vorhanden). In einer Bucket-Richtlinie können Sie eine Bedingung hinzufügen, um diesen Wert zu überprüfen, wie im folgenden Beispiel gezeigt.

Die Beispielrichtlinie verweigert jede Amazon-S3-Operation am */taxdocuments*-Ordner in dem *DOC-EXAMPLE-BUCKET*-Bucket, wenn die Anforderung nicht über die MFA authentifiziert wird. Weitere Informationen über die MFA finden Sie unter Using Multi-Factor Authentication (MFA) (Verwenden der Multi-Factor-Authentifizierung (MFA)) in AWS im IAM-Benutzerhandbuch.

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/taxdocuments/*",
        "Condition": { "Null": { "aws:MultiFactorAuthAge": true }}
      }
```

```
    ]
}
```

Die `Null`-Bedingung im `Condition`-Block wird zu `true` ausgewertet, wenn der `aws:MultiFactorAuthAge`-Bedingungsschlüsselwert null ist, d. h., die temporären Anmeldeinformationen in der Anforderung wurden ohne ein MFA-Gerät erstellt.

Die folgende Bucket-Richtlinie ist eine Erweiterung der vorhergehenden Bucket-Richtlinie. Sie beinhaltet zwei Richtlinienanweisungen. Eine Anweisung gewährt jedem die `s3:GetObject`-Berechtigung für einen Bucket (*`DOC-EXAMPLE-BUCKET`*). Eine weitere Anweisung schränkt den Zugriff auf den *`DOC-EXAMPLE-BUCKET/taxdocuments`*-Ordner im Bucket weiter ein, indem sie eine MFA erzwingt.

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/taxdocuments/*",
        "Condition": { "Null": { "aws:MultiFactorAuthAge": true } }
      },
      {
        "Sid": "",
        "Effect": "Allow",
        "Principal": "*",
        "Action": ["s3:GetObject"],
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      }
    ]
}
```

Optional können Sie eine numerische Bedingung verwenden, um die Gültigkeitsdauer des `aws:MultiFactorAuthAge`-Schlüssels zu beschränken. Die Dauer, die Sie für den `aws:MultiFactorAuthAge`-Schlüssel angeben, ist unabhängig von der Lebensdauer der temporären Sicherheitsanmeldeinformationen, die für die Authentifizierung der Anforderung verwendet wurden.

Beispielsweise überprüft die folgende Bucket-Richtlinie zusätzlich zur geforderten MFA-Authentifizierung auch, vor wie langer Zeit die temporäre Sitzung erstellt wurde. Die Richtlinie verweigert jede Operation, wenn der `aws:MultiFactorAuthAge`-Schlüsselwert angibt, dass die temporäre Sitzung vor mehr als einer Stunde erstellt wurde (3600 Sekunden).

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/taxdocuments/*",
        "Condition": {"Null": {"aws:MultiFactorAuthAge": true }}
      },
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/taxdocuments/*",
        "Condition": {"NumericGreaterThan": {"aws:MultiFactorAuthAge": 3600 }}
       },
       {
         "Sid": "",
         "Effect": "Allow",
         "Principal": "*",
         "Action": ["s3:GetObject"],
         "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
       }
    ]
 }
```

Verwenden von IAM-Benutzer- und -rollenrichtlinien

Sie können IAM-Benutze- oder Rollenrichtlinien erstellen und konfigurieren, um den Zugriff auf Amazon S3 zu kontrollieren. Benutzer- oder Rollenrichtlinien verwenden die JSON-basierte Sprache der Zugriffsrichtlinie.

Dieser Abschnitt enthält mehrere IAM-Benutzer- und -rollenrichtlinien zum Kontrollieren des Benutzerzugriffs auf Amazon S3. Beispiele für Bucket-Richtlinien finden Sie unter Verwenden von Bucket-Richtlinien. Informationen zur Sprache der Zugriffsichtlinie finden Sie unter Richtlinien und Berechtigungen in Amazon S3.

Themen

- Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien
- Beispiele für Benutzer- und Rollenrichtlinien

Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien

Diese schrittweise Anleitung erklärt, wie Benutzerberechtigungen in Amazon S3 funktionieren. In diesem Beispiel erstellen Sie einen Bucket mit Ordnern. Anschließend erstellen Sie AWS Identity and Access Management IAM-Benutzer in Ihrem AWS-Konto und erteilen diesen Benutzern inkrementelle Berechtigungen für Ihren Amazon S3-Bucket und die darin enthaltenen Ordner.

Themen

- Grundlagen zu Buckets und Ordnern
- Walkthrough-Übersicht
- Vorbereitung auf den Walkthrough
- Schritt 1: Erstellen eines Buckets
- Schritt 2: Erstellen von IAM-Benutzern und einer Gruppe
- Schritt 3: Überprüfen Sie, dass die IAM-Benutzer über keine Berechtigungen verfügen
- Schritt 4: Erteilen von Berechtigungen auf Gruppenebene
- Schritt 5: Der IAM-Benutzerin Alice spezifische Berechtigungen erteilen
- Schritt 6: Dem IAM-Benutzer Bob spezifische Berechtigungen erteilen
- Schritt 7: Absichern des Ordners „Private“
- Schritt 8: Bereinigen
- Zugehörige Ressourcen

Grundlagen zu Buckets und Ordnern

Das Amazon-S3-Datenmodell ist eine flache Struktur: Sie erstellen einen Bucket und der Bucket speichert Objekte. Es gibt keine Hierarchie für Unter-Buckets oder Unterordner. Sie können aber

eine Ordnerhierarchie nachbilden. Tools, wie die Amazon-S3-Konsole, können eine Ansicht dieser logischen Ordner und Unterordner in Ihrem Bucket präsentieren.

Die Konsole zeigt, dass sich in einem Bucket mit dem Namen `companybucket` die drei Ordner `Private`, `Development` und `Finance` und das Objekt `s3-dg.pdf` befinden. Die Konsole verwendet die Objektnamen (Schlüssel), um eine logische Hierarchie mit Ordnern und Unterordnern zu erzeugen. Betrachten Sie die folgenden Beispiele:

- Wenn Sie den Ordner `Development` anlegen, erstellt die Konsole ein Objekt mit dem Schlüssel `Development/`. Beachten Sie den als Trennzeichen dienenden abschließenden Schrägstrich (`/`).
- Wenn Sie ein Objekt mit dem Namen `Projects1.xls` in den Ordner `Development` hochladen, lädt die Konsole das Objekt hoch und weist ihm den Schlüssel `Development/Projects1.xls` zu.

  Im Schlüssel ist `Development` das Präfix und `/` ist das Trennzeichen. Die Amazon-S3-API unterstützt in ihren Vorgängen Präfixe und Separatoren. Beispielsweise können Sie eine Liste aller Objekte in einem Bucket mit einem bestimmten Präfix und Separator erhalten. Wenn Sie in der Konsole den Ordner `Development` öffnen, listet die Konsole die Objekte im betreffenden Ordner auf. Im folgenden Beispiel enthält der Ordner `Development` ein Objekt.

  Bildschirm der Konsole mit einer Hierarchie der Buckets, Ordner und Objekte.

  Wenn die Konsole den Ordner `Development` im Bucket `companybucket` auflistet, sendet sie eine Anforderung an Amazon S3, in der das Präfix `Development` und das Trennzeichen `/` in der Anforderung angegeben werden. Die Antwort der Konsole sieht genauso aus, wie eine Ordnerliste im Dateisystem Ihres Computers. Das vorherige Beispiel zeigt, dass der Bucket `companybucket` ein Objekt mit dem Schlüssel `Development/Projects1.xls` enthält.

Die Konsole verwendet Objektschlüssel, um eine logische Hierarchie abzuleiten. Amazon S3 hat keine physische Hierarchie; es hat nur Buckets, die Objekte in einer Flatfile-Struktur enthalten. Wenn Sie Objekte mit der Amazon-S3-API erstellen, können Sie Objektschlüssel verwenden, die eine logische Hierarchie implizieren. Wenn Sie eine logische Hierarchie von Objekten erstellen, können Sie den Zugriff auf einzelne Ordner verwalten, wie in dieser exemplarischen Anleitung veranschaulicht.

Bevor Sie beginnen, müssen Sie mit dem Konzept des Bucket-Inhalts auf Stammebene vertraut sein. Angenommen, Ihr Bucket `companybucket` enthält die folgenden Objekte:

- `Private/privDoc1.txt`
- `Private/privDoc2.zip`
- `Development/project1.xls`
- `Development/project2.xls`
- `Finance/Tax2011/document1.pdf`
- `Finance/Tax2011/document2.pdf`
- `s3-dg.pdf`

Diese Objektschlüssel erzeugen eine logische Hierarchie mit `Private`, `Development` und `Finance` als Stammebenen-Ordner und `s3-dg.pdf` als Stammebenen-Objekt. Wenn Sie den Bucket-Namen in der Amazon-S3-Konsole auswählen, erscheinen die Elemente der Stammebene. Die Konsole zeigt die Präfixe der obersten Ebene (`Private/`, `Development/` und `Finance/`) als Stammebenen-Ordner. Der Objektschlüssel `s3-dg.pdf` hat kein Präfix und erscheint daher als Stammebenen-Element.

Walkthrough-Übersicht

In dieser Anleitung erstellen Sie einen Bucket mit drei Ordnern (`Private`, `Development` und `Finance`).

Sie haben zwei User, Alice und Bob. Sie möchten, dass Alice nur auf den Ordner `Development` und Bob nur auf den Ordner `Finance` zugreift. Sie möchten, das der Inhalt des Ordners `Private` privat bleibt. In der Anleitung verwalten Sie den Zugriff durch Erstellen von IAM-Benutzern (im Beispiel werden die Benutzernamen Alice und Bob verwendet) und gewähren ihnen die erforderlichen Berechtigungen.

IAM unterstützt auch die Erstellung von Benutzergruppen und von Berechtigungen auf Gruppenebene, die für alle Benutzer in der Gruppe gelten. Dies hilft Ihnen bei der Verwaltung der Berechtigungen. In dieser Übung benötigen sowohl Alice als auch Bob einige gemeinsame Berechtigungen. Sie erstellen also eine Gruppe mit dem Namen `Consultants` und fügen dann Alice und Bob der Gruppe hinzu. Sie erteilen zunächst Berechtigungen, indem Sie der Gruppe eine Gruppenrichtlinie zuweisen. Fügen Sie dann benutzerspezifische Berechtigungen durch Zuweisen von Richtlinien zu bestimmten Benutzern hinzu.

> **Note**
>
> Die Anleitung verwendet `companybucket` als Bucket-Namen, Alice und Bob als IAM-Benutzer und `Consultants` als Gruppenname. Da Bucket-Namen in Amazon S3 global eindeutig sein müssen, müssen Sie den Bucket-Namen durch einen von Ihnen erstellten Namen ersetzen.

Vorbereitung auf den Walkthrough

In diesem Beispiel verwenden Sie Ihre - AWS-Konto Anmeldeinformationen, um IAM-Benutzer zu erstellen. Zu Beginn haben diese Benutzer keine Berechtigungen. Sie gewähren diesen Benutzern nach und nach Berechtigungen, damit sie spezifische Amazon-S3-Aktionen ausführen können. Um diese Berechtigungen zu testen, melden Sie sich mit den Anmeldeinformationen eines jeden Benutzers bei der Konsole an. Wenn Sie als AWS-Konto Eigentümer schrittweise Berechtigungen erteilen und Berechtigungen als IAM-Benutzer testen, müssen Sie sich jedes Mal mit unterschiedlichen Anmeldeinformationen anmelden und abmelden. Sie können diesen Test auch mit einem Browser durchführen, das Verfahren schreitet aber schneller fort, wenn Sie zwei verschiedene Browser verwenden. Verwenden Sie einen Browser, um eine Verbindung mit dem AWS Management Console mit Ihren - AWS-Konto Anmeldeinformationen herzustellen, und einen anderen, um eine Verbindung mit den IAM-Benutzer-Anmeldeinformationen herzustellen.

Um sich bei der AWS Management Console mit Ihren AWS-Konto -Anmeldeinformationen anzumelden, gehen Sie zu https://console.aws.amazon.com/. Ein IAM-Benutzer kann sich nicht über denselben Link anmelden. Ein IAM-Benutzer muss eine IAM-aktivierte Anmeldeseite verwenden. Als Kontoinhaber können Sie diesen Link Ihren Benutzern bereitstellen.

Weitere Informationen zu IAM finden Sie unter Die AWS Management Console Anmeldeseite im IAM-Benutzerhandbuch.

So stellen Sie einen Anmeldelink für IAM-Benutzer bereit:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich IAM Dashboard aus.
3. Notieren Sie die URL unter IAM users sign in link (Anmeldelink für IAM-Benutzer):. Sie geben diesen Link an die IAM-Benutzer weiter, damit sie sich mit ihrem IAM-Benutzernamen und -Passwort in der Konsole anmelden können.

Schritt 1: Erstellen eines Buckets

In diesem Schritt melden Sie sich bei der Amazon S3-Konsole mit Ihren - AWS-Konto Anmeldeinformationen an, erstellen einen Bucket, fügen dem Bucket Ordner (`DevelopmentFinance`, und `Private`) hinzu und laden ein oder zwei Beispieldokumente in jeden Ordner hoch.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Erstellen Sie einen Bucket.

   step-by-step Anweisungen finden Sie unter Erstellen eines Buckets.
3. Laden Sie ein Dokument in den Bucket hoch.

   Diese Übung geht davon aus, dass sich das Dokument `s3-dg.pdf` auf der Stammebene dieses Buckets befindet. Wenn Sie andere Dokumente hochladen, ändern Sie ihren Dateinamen in `s3-dg.pdf`.
4. Fügen Sie drei Ordner mit dem Namen `Private`, `Finance` and `Development` zum Bucket hinzu.

   step-by-step Anweisungen zum Erstellen eines Ordners finden Sie unter Organisieren von Objekten in der Amazon S3-Konsole mithilfe von Ordnern> im Benutzerhandbuch für Amazon Simple Storage Service.
5. Laden Sie ein oder zwei Dokumente in jeden Ordner hoch.

   Für diese Übung wird angenommen, dass Sie einige Dokumente in jeden Ordner hochgeladen haben, sodass der Bucket Objekte mit den folgenden Schlüsseln enthält:

   - `Private/privDoc1.txt`
   - `Private/privDoc2.zip`
   - `Development/project1.xls`
   - `Development/project2.xls`
   - `Finance/Tax2011/document1.pdf`
   - `Finance/Tax2011/document2.pdf`
   - `s3-dg.pdf`

step-by-step Anweisungen finden Sie unter Objekte hochladen.

Schritt 2: Erstellen von IAM-Benutzern und einer Gruppe

Verwenden Sie jetzt die IAM-Konsole, um zwei IAM-Benutzer, Alice und Bob, zu Ihrem hinzuzufügen AWS-Konto. step-by-step Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.

Erstellen Sie auch eine administrative Gruppe mit dem Namen `Consultants`. Fügen Sie dann beide Benutzer zur Gruppe hinzu. step-by-step Anweisungen finden Sie unter Erstellen von IAM-Benutzergruppen.

> **Warning**
>
> Wenn Sie die Benutzer und die Gruppe erstellen, fügen Sie keine Richtlinien an, die diesen Benutzern Berechtigungen erteilen. Anfänglich haben diese Benutzer keine Berechtigungen. In den folgenden Abschnitten gewähren Sie nach und nach Berechtigungen. Sie müssen zunächst sicherstellen, dass Sie diesen IAM-Benutzern Passwörter zugewiesen haben. Sie verwenden diese Benutzer-Anmeldeinformationen zum Testen der Amazon-S3-Aktionen und zum Überprüfen, ob die Berechtigungen wie erwartet funktionieren.

step-by-step Anweisungen zum Erstellen eines neuen IAM-Benutzers finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch. Wenn Sie die Benutzer für diese Anleitung erstellen, wählen Sie AWS Management Console -Zugriff aus und deaktivieren Sie Programmgesteuerter Zugriff.

step-by-step Anweisungen zum Erstellen einer Administratorgruppe finden Sie unter Erstellen Ihres ersten IAM-Administratorbenutzers und Ihrer ersten Administratorgruppe im IAM-Benutzerhandbuch.

Schritt 3: Überprüfen Sie, dass die IAM-Benutzer über keine Berechtigungen verfügen

Wenn Sie zwei Browser verwenden, können Sie jetzt den zweiten Browser verwenden, um in der Konsole einen IAM-Benutzer mit seinen Anmeldeinformationen anzumelden.

1. Melden Sie sich über den Anmeldelink des IAM-Benutzers (siehe So stellen Sie einen Anmeldelink für IAM-Benutzer bereit:) bei der AWS Management Console unter Verwendung einer der IAM-Benutzeranmeldeinformationen an.

2. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   Die folgende Konsolenmeldung informiert Sie darüber, dass der Zugang verweigert wurde.
   Screenshot der Konsole mit einer Fehlermeldung bei nicht genehmigtem Zugriff.

Nun können Sie damit beginnen, den Benutzern schrittweise Berechtigungen zu erteilen. Sie weisen zunächst eine Gruppenrichtlinie zu, die beiden Benutzern die erforderlichen Berechtigungen gewährt.

Schritt 4: Erteilen von Berechtigungen auf Gruppenebene

Sie möchten den Benutzern Folgendes ermöglichen:

- Auflisten aller Buckets, die dem übergeordneten Konto gehören. Um das zu tun, müssen Bob und Alice die Berechtigung für die Aktion `s3:ListAllMyBuckets` besitzen.
- Auflisten aller Elemente, Ordner und Objekte auf Stammebene im Bucket `companybucket`. Um das zu tun, müssen Bob und Alice die Berechtigung für die Aktion `s3:ListBucket` im Bucket `companybucket` besitzen.

Zuerst erstellen Sie eine Richtlinie, die diese Berechtigungen gewährt, und dann weisen Sie sie der Gruppe `Consultants` zu.

Schritt 4.1: Erteilen der Berechtigung zum Auflisten aller Buckets

In diesem Schritt erstellen Sie eine verwaltete Richtlinie, die den Benutzern die Mindestberechtigungen für die Auflistung aller Buckets des übergeordneten Kontos erteilt. Dann weisen Sie die Richtlinie der Gruppe `Consultants` zu. Wenn Sie einem Benutzer oder einer Benutzergruppe die verwaltete Richtlinie zuordnen, erhält der Benutzer oder die Gruppe die Berechtigung, alle Buckets des übergeordneten AWS-Konto aufzulisten.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   > **Note**
   >
   > Da Sie Benutzerberechtigungen erteilen, melden Sie sich mit Ihren - AWS-Konto Anmeldeinformationen an, nicht als IAM-Benutzer.

2. Erstellen Sie die verwaltete Richtlinie.

a. Wählen Sie links im Navigationsbereich Policies (Richtlinien) und dann Create Policy (Richtlinie erstellen) aus.

b. Wählen Sie den Tab JSON.

c. Kopieren Sie die folgende Zugriffsrichtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGroupToSeeBucketListInTheConsole",
      "Action": ["s3:ListAllMyBuckets"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::*"]
    }
  ]
}
```

Eine Richtlinie ist ein JSON-Dokument. Im Dokument ist ein `Statement` ein Array von Objekten, die jeweils eine Berechtigung unter Verwendung einer Sammlung von Namenswertpaaren beschreiben. Die vorangegangene Richtlinie beschreibt eine bestimmte Berechtigung. Die `Action` definiert den Zugriffstyp. In der Richtlinie ist `s3:ListAllMyBuckets` eine vordefinierte Amazon-S3-Aktion. Diese Aktion umfasst die Operation Amazon S3 GET Service, die eine Liste aller Buckets des authentifizierten Absenders zurückgibt. Der Wert des `Effect`-Elements bestimmt, ob die spezifische Berechtigung gewährt oder verweigert wird.

d. Wählen Sie Review policy (Richtlinie überprüfen) aus. Geben Sie auf der nächsten Seite in das Feld Name `AllowGroupToSeeBucketListInTheConsole` ein und wählen Sie dann Create policy (Richtlinie erstellen).

> **Note**
>
> Der Eintrag Summary (Übersicht) enthält eine Nachricht, die angibt, dass die Richtlinie keinerlei Berechtigungen gewährt. Für diese Anleitung können Sie diese Nachricht getrost ignorieren.

3. Weisen Sie die von `AllowGroupToSeeBucketListInTheConsole` verwaltete Richtlinie, die Sie erstellt haben, der Gruppe `Consultants` zu.

   step-by-step Anweisungen zum Anfügen einer verwalteten Richtlinie finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

   Sie fügen die Richtliniendokumente in der IAM-Konsole den IAM-Benutzern und Gruppen hinzu. Da Sie möchten, dass beide Benutzer die Buckets auflisten können, weisen Sie die Richtlinie der Gruppe zu.
4. Die Berechtigung testen.

   a. Verwenden Sie den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmeldelink für IAM-Benutzer bereit:) beim Anmelden in der Konsole mit den beiden verschiedenen IAM-Anmeldeinformationen.

   b. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

      Die Konsole sollte nun alle Buckets auflisten, nicht aber die Objekte in den jeweiligen Buckets.
      Screenshot der Konsole mit einer Liste der Buckets.

Schritt 4.2: Benutzern gestatten, dass sie den Bucket-Inhalt auf Stammebene auflisten

Als Nächstes gestatten Sie allen Benutzern in der Gruppe `Consultants`, Elemente im Bucket `companybucket` aufzulisten. Wenn ein Benutzer den Unternehmensbucket in der Amazon-S3-Konsole auswählt, werden die Elemente in der Stammebene des Buckets angezeigt.
Screenshot der Konsole mit dem Inhalt von companybucket.

> **Note**
>
> Dieses Beispiel verwendet `companybucket` zur Veranschaulichung. Sie müssen den Namen des Bucket verwenden, den Sie erstellt haben.

Um zu verstehen, welche Anforderung die Konsole an Amazon S3 sendet, wenn Sie einen Bucket-Namen auswählen, welche Antwort Amazon S3 zurückgibt und wie die Konsole die Antwort interpretiert, ist es notwendig, etwas tiefer in die Materie einzudringen.

Wenn Sie auf einen Bucket-Namen klicken, sendet die Konsole die Anforderung GET Bucket (List Objects) an Amazon S3. Diese Anforderung enthält die folgenden Parameter:

- Der Parameter `prefix` mit einer leeren Zeichenfolge als Wert.
- Der Parameter `delimiter` mit `/` als Wert.

Es folgt ein Beispiel einer Anforderung.

```
GET ?prefix=&delimiter=/ HTTP/1.1
Host: companybucket.s3.amazonaws.com
Date: Wed, 01 Aug  2012 12:00:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:xQE0diMbLRepdf3YB+FIEXAMPLE=
```

Amazon S3 gibt eine Antwort zurück, die das folgende <ListBucketResult/>-Element enthält:

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>companybucket</Name>
  <Prefix></Prefix>
  <Delimiter>/</Delimiter>
   ...
  <Contents>
    <Key>s3-dg.pdf</Key>
    ...
  </Contents>
  <CommonPrefixes>
    <Prefix>Development/</Prefix>
  </CommonPrefixes>
  <CommonPrefixes>
    <Prefix>Finance/</Prefix>
  </CommonPrefixes>
  <CommonPrefixes>
    <Prefix>Private/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

Das Schlüsselobjekt `s3-dg.pdf` enthält nicht das Schrägstrich-Trennzeichen (`/`), und Amazon S3 gibt den Schlüssel im <Contents>-Element zurück. Alle anderen Schlüssel in unserem Beispiel-Bucket enthalten jedoch das `/`-Trennzeichen. Amazon S3 gruppiert diese Schlüssel und gibt ein einziges <CommonPrefixes>-Element für jeden der verschiedenen Präfix-Werte `Development/`, `Finance/` und `Private/` zurück, eine Unterzeichenfolge vom Anfang dieser Schlüssel bis zum ersten Auftreten des angegebenen `/`-Trennzeichens.

Die Konsole interpretiert dieses Ergebnis und zeigt die Elemente auf der Stammebene als drei Ordner und einen Objektschlüssel an.
Screenshot der Konsole mit dem Inhalt von companybucket mit drei Ordnern und einer PDF-Datei.

Wenn nun Bob oder Alice den Ordner Development öffnen, sendet die Konsole die Anforderung GET Bucket (List Objects) an Amazon S3, wobei die Parameter `prefix` und `delimiter` auf die folgenden Werte eingestellt sind:

- Der Parameter `prefix` mit dem Wert `Development/`.
- Der Parameter `delimiter` mit dem Wert "`/`".

In der Antwort gibt Amazon S3 die Objektschlüssel mit dem angegebenen Präfix zurück.

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>companybucket</Name>
  <Prefix>Development</Prefix>
  <Delimiter>/</Delimiter>
   ...
  <Contents>
    <Key>Project1.xls</Key>
    ...
  </Contents>
  <Contents>
    <Key>Project2.xls</Key>
    ...
  </Contents>
</ListBucketResult>
```

Die Konsole zeigt die Objektschlüssel an.
Screenshot der Konsole mit dem Development-Ordner mit zwei XLS-Dateien.

Kehren Sie nun wieder zu dem Vorgang zurück, Benutzern die Berechtigung zum Auflisten von Elementen auf der Stammebene zu erteilen. Damit der Bucket-Inhalt aufgelistet werden kann, müssen die Benutzer die Berechtigung zum Aufruf der `s3:ListBucket`-Aktion besitzen, wie in der folgenden Richtlinienanweisung dargestellt. Um sicherzustellen, dass sie nur den Inhalt auf der Stammebene sehen, können Sie als Bedingung hinzufügen, dass Benutzer in der Anforderung ein leeres `prefix` angeben müssen – d. h., dass es ihnen nicht gestattet sein soll, auf einen der Stammebenen-Ordner doppelzuklicken. Schließlich fügen Sie noch eine Bedingung hinzu, die den Zugriff im Ordnerstil dadurch vorschreibt, dass Benutzeranforderungen den Parameter `delimiter` mit dem Wert "`/`" enthalten müssen.

```
{
  "Sid": "AllowRootLevelListingOfCompanyBucket",
  "Action": ["s3:ListBucket"],
  "Effect": "Allow",
  "Resource": ["arn:aws:s3:::companybucket"],
  "Condition":{
         "StringEquals":{
             "s3:prefix":[""], "s3:delimiter":["/"]
                       }
              }
}
```

Wenn Sie einen Bucket in der Amazon S3-Konsole auswählen, sendet die Konsole zuerst die Anforderung GET Bucket location, um die zu finden, AWS-Region in der der Bucket bereitgestellt wird. Dann verwendet die Konsole den regionsspezifischen Endpunkt für den Bucket, um die Anforderung GET Bucket (List Objects) zu senden. Wenn Benutzer die Konsole verwenden sollen, müssen Sie ihnen folglich die Berechtigung für die Aktion `s3:GetBucketLocation` gewähren, wie in der folgenden Richtlinienanweisung veranschaulicht.

```
{
   "Sid": "RequiredByS3Console",
   "Action": ["s3:GetBucketLocation"],
   "Effect": "Allow",
   "Resource": ["arn:aws:s3:::*"]
}
```

Den Benutzern gestatten, den Bucket-Inhalt auf Stammebene aufzulisten

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   Verwenden Sie Ihre - AWS-Konto Anmeldeinformationen, nicht die eines IAM-Benutzers, um sich bei der Konsole anzumelden.
2. Ersetzen Sie die vorhandene `AllowGroupToSeeBucketListInTheConsole`-verwaltete Richtlinie, die der Gruppe `s3:ListBucket` zugeordnet ist, durch die folgende Richtlinie, durch die die Aktion `Consultants` ebenfalls gestattet wird. Denken Sie daran, den Namen *`companybucket`* in der Richtlinie `Resource` durch den Namen Ihres Buckets zu ersetzen.

   step-by-step Anweisungen finden Sie unter Bearbeiten von IAM-Richtlinien im IAM-Benutzerhandbuch. Wenn Sie die step-by-step Anweisungen befolgen, befolgen Sie unbedingt

die Schritte zum Anwenden Ihrer Änderungen auf alle Prinzipal-Entitäten, denen die Richtlinie angefügt ist.

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Sid":
 "AllowGroupToSeeBucketListAndAlsoAllowGetBucketLocationRequiredForListBucket",
       "Action": [ "s3:ListAllMyBuckets", "s3:GetBucketLocation" ],
       "Effect": "Allow",
       "Resource": [ "arn:aws:s3:::*"  ]
     },
     {
       "Sid": "AllowRootLevelListingOfCompanyBucket",
       "Action": ["s3:ListBucket"],
       "Effect": "Allow",
       "Resource": ["arn:aws:s3:::companybucket"],
       "Condition":{
             "StringEquals":{
                    "s3:prefix":[""], "s3:delimiter":["/"]
                           }
                  }
     }
  ]
}
```

3. Testen Sie die aktualisierten Berechtigungen.

   a. Verwenden Sie den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmeldelink für IAM-Benutzer bereit:) für die Anmeldung in der AWS Management Console.

      Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   b. Wählen Sie den von Ihnen erstellten Bucket aus. Die Konsole zeigt die Bucket-Elemente auf Stammebene an. Wenn Sie Ordner im Bucket auswählen, können Sie den Ordnerinhalt nicht sehen, da Sie diese Berechtigungen noch nicht gewährt haben.
      Screenshot der Konsole mit dem Unternehmensbucket mit drei Ordnern.

Dieser Test ist erfolgreich, wenn Benutzer die Amazon-S3-Konsole verwenden. Wenn Sie einen Bucket in der Konsole auswählen, sendet die Konsolenimplementierung eine Anforderung mit dem

Parameter `prefix` mit einer leeren Zeichenfolge als Wert und dem Parameter `delimiter` mit "`/`" als Wert.

Schritt 4.3: Übersicht über die Gruppenrichtlinie

Die Wirkung der von Ihnen hinzugefügten Gruppenrichtlinie ist, dass die IAM-Benutzer Alice und Bob über die folgenden Mindestberechtigungen verfügen:

- Auflisten aller Buckets, die dem übergeordneten Konto gehören.
- Ansicht der Elemente auf Stammebene im Bucket `companybucket`.

Die Benutzer können jedoch noch nicht sehr viele Aktionen ausführen. Als Nächstes erteilen Sie die folgenden benutzerspezifischen Berechtigungen:

- Erlauben Sie Bob, Objekte im Ordner `Development` aufzurufen und abzulegen.
- Erlauben Sie Bob, Objekte im Ordner `Finance` aufzurufen und abzulegen.

Für benutzerspezifische Berechtigungen fügen Sie eine Richtlinie zum spezifischen Benutzer hinzu, nicht für die Gruppe. Im folgenden Abschnitt erteilen Sie Alice die Berechtigung, mit dem Ordner `Development` zu arbeiten. Sie können die Schritte wiederholen, um Bob eine ähnliche Berechtigung für das Arbeiten im Ordner `Finance` zu erteilen.

Schritt 5: Der IAM-Benutzerin Alice spezifische Berechtigungen erteilen

Nun gewähren Sie Alice zusätzliche Berechtigungen, damit sie den Inhalt des Ordners `Development` aufrufen und Objekte in diesem Ordner ablegen kann.

Schritt 5.1: Der IAM-Benutzerin Alice die Berechtigung erteilen, den Inhalt des Development-Ordners aufzulisten

Damit Alice den Inhalt des Ordners `Development` auflisten kann, müssen Sie der Benutzerin Alice eine Richtlinie zuweisen, die die Berechtigung für die Aktion `s3:ListBucket` für den Bucket `companybucket` erteilt, unter der Voraussetzung, dass die Anforderung das Präfix `Development/` enthält. Da diese Richtlinie nur auf die Benutzerin Alice angewendet werden soll, verwenden Sie eine Inline-Richtlinie. Weitere Informationen zu Inline-Richtlinien finden Sie unter Verwaltete Richtlinien und Inline-Richtlinien im IAM-Benutzerhandbuch.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

Verwenden Sie Ihre - AWS-Konto Anmeldeinformationen, nicht die eines IAM-Benutzers, um sich bei der Konsole anzumelden.

2. Erstellen Sie eine Inline-Richtlinie, um der Benutzerin Alice die Berechtigung zu erteilen, den Inhalt des Ordners `Development` aufzulisten.

   a. Wählen Sie im Navigationsbereich auf der linken Seite Users (Benutzer) aus.

   b. Klicken Sie auf den Benutzernamen Alice.

   c. Wählen Sie auf der Benutzerdetailseite den Tab Permissions (Berechtigungen) und dann Add inline policy (Inline-Richtlinie hinzufügen).

   d. Wählen Sie den Tab JSON.

   e. Kopieren Sie die folgende Richtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
      "Sid": "AllowListBucketIfSpecificPrefixIsIncludedInRequest",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::companybucket"],
      "Condition":{  "StringLike":{"s3:prefix":["Development/*"] }
       }
    }
  ]
}
```

   f. Wählen Sie Review policy (Richtlinie überprüfen) aus. Geben Sie auf der nächsten Seite in das Feld Name einen Namen ein und wählen Sie dann Create policy (Richtlinie erstellen).

3. Testen Sie die geänderten Berechtigungen für Alice:

   a. Verwenden Sie den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmeldelink für IAM-Benutzer bereit:) für die Anmeldung in der AWS Management Console.

   b. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   c. Prüfen Sie in der Amazon-S3-Konsole, ob Alice die Liste der Objekte im Ordner `Development/` im Bucket sehen kann.

      Wenn die Benutzerin den Ordner `/Development` auswählt, um die Liste der darin enthaltenen Objekte anzuzeigen, sendet die Amazon-S3-Konsole die Anforderung

`ListObjects` zusammen mit dem Präfix `/Development` an Amazon S3. Weil die Benutzerin die Erlaubnis besitzt, die Objektliste mit dem Präfix `Development` und dem Trennzeichen `/` zu sehen, gibt Amazon S3 die Objektliste mit dem Schlüsselpräfix `Development/` zurück, und die Konsole gibt die Liste aus.
Screenshot der Konsole mit dem Development-Ordner mit zwei XLS-Dateien.

Schritt 5.2: Der IAM-Benutzerin Alice die Berechtigung erteilen, auf die Objekte im Development-Ordner zuzugreifen und Objekte darin abzulegen

Damit Alice Objekte im Ordner `Development` ablegen und aufrufen kann, benötigt sie die Berechtigung für die Aktionen `s3:GetObject` und `s3:PutObject`. Die folgenden Richtlinienanweisungen räumen diese Berechtigungen ein, vorausgesetzt die Anforderung enthält den Parameter `prefix` mit dem Wert `Development/`.

```
{
    "Sid":"AllowUserToReadWriteObjectData",
    "Action":["s3:GetObject", "s3:PutObject"],
    "Effect":"Allow",
    "Resource":["arn:aws:s3:::companybucket/Development/*"]
 }
```

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   Verwenden Sie Ihre - AWS-Konto Anmeldeinformationen, nicht die eines IAM-Benutzers, um sich bei der Konsole anzumelden.
2. Bearbeiten Sie die Inline-Richtlinie, die Sie im vorherigen Schritt erstellt haben.

   a. Wählen Sie im Navigationsbereich auf der linken Seite Users (Benutzer) aus.

   b. Klicken Sie auf den Benutzernamen Alice.

   c. Wählen Sie auf der Benutzerdetailseite den Tab Permissions (Berechtigungen) aus und erweitern Sie den Bereich Inline Policies (Inline-Richtlinien).

   d. Wählen Sie neben dem Namen der im vorherigen Schritt erstellten Richtlinie Edit Policy (Richtlinie bearbeiten) aus.

   e. Kopieren Sie die folgende Richtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein, wobei die vorhandene Richtlinie ersetzt wird.

```
{
     "Version": "2012-10-17",
     "Statement":[
      {
         "Sid":"AllowListBucketIfSpecificPrefixIsIncludedInRequest",
         "Action":["s3:ListBucket"],
         "Effect":"Allow",
         "Resource":["arn:aws:s3:::companybucket"],
         "Condition":{
            "StringLike":{"s3:prefix":["Development/*"]
            }
         }
      },
      {
        "Sid":"AllowUserToReadWriteObjectDataInDevelopmentFolder",
        "Action":["s3:GetObject", "s3:PutObject"],
        "Effect":"Allow",
        "Resource":["arn:aws:s3:::companybucket/Development/*"]
      }
   ]
}
```

3. Testen Sie die aktualisierte Richtlinie:

   a. Verwenden Sie den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmeldelink für IAM-Benutzer bereit:) für die Anmeldung in der AWS Management Console.

   b. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   c. Prüfen Sie in der Amazon-S3-Konsole, ob Alice nun im Ordner `Development` ein Objekt hinzufügen oder herunterladen kann.

Schritt 5.3: Der IAM-Benutzerin Alice die Berechtigung für den Zugriff auf alle anderen Ordner im Bucket ausdrücklich verweigern

Die Benutzerin kann jetzt den Inhalt auf Stammebene im Bucket `companybucket` auflisten. Sie kann auch Objekte im Ordner `Development` aufrufen und ablegen. Wenn Sie die Zugriffsberechtigungen weiter verbessern möchten, können Sie Alice den Zugriff auf andere Ordner im Bucket explizit verweigern. Wenn es irgendeine andere Richtlinie (Bucket-Richtlinie oder ACL) gibt, die Alice den Zugriff auf andere Ordner im Bucket gewährt, überschreibt diese explizite Zugriffsverweigerung diese Berechtigungen.

Sie können die folgende Anweisung zur Benutzerrichtlinie für Alice hinzufügen, die von allen Anfragen von Alice an Amazon S3 erfordert, dass der Parameter `prefix` enthalten ist, dessen Wert entweder `Development/*` oder eine leere Zeichenfolge ist.

```
{
   "Sid": "ExplicitlyDenyAnyRequestsForAllOtherFoldersExceptDevelopment",
   "Action": ["s3:ListBucket"],
   "Effect": "Deny",
   "Resource": ["arn:aws:s3:::companybucket"],
   "Condition":{  "StringNotLike": {"s3:prefix":["Development/*",""] },
                  "Null"         : {"s3:prefix":false }
    }
}
```

Beachten Sie, dass im Block `Condition` zwei bedingte Ausdrücke enthalten sind. Das Ergebnis dieser bedingten Ausdrücke wird durch das logische `AND` verknüpft. Wenn beide Bedingungen wahr sind, ist das Ergebnis der bedingten Ausdrücke wahr. Da der Wert für `Effect` in dieser Richtlinie `Deny` lautet, wenn `Condition` als "true" bewertet wird, können Benutzer die angegebene `Action` nicht durchführen.

- Der bedingte Ausdruck `Null` stellt sicher, dass die Anforderung von Alice den Parameter `prefix` enthält.

  Der Parameter `prefix` erfordert einen ordnerartigen Zugriff. Wenn Sie eine Anfrage ohne den Parameter `prefix` senden, gibt Amazon S3 alle Objektschlüssel zurück.

  Wenn die Anforderung den Parameter `prefix` mit einem Nullwert enthält, wird der Ausdruck als wahr ausgewertet, womit die gesamte Bedingung `Condition` wahr ist. Sie müssen eine leere Zeichenfolge für den Parameter `prefix` gestatten. Erinnern Sie sich an die vorangegangene Diskussion. Die leere Zeichenfolge lässt zu, dass Alice Bucket-Elemente auf Stammebene abrufen kann, wie es die Konsole in der vorhergehenden Diskussion macht. Weitere Informationen finden Sie unter Schritt 4.2: Benutzern gestatten, dass sie den Bucket-Inhalt auf Stammebene auflisten.
- Der bedingte Ausdruck `StringNotLike` stellt sicher, dass die Anforderung scheitert, wenn der angegebene Wert des Parameters `prefix` nicht `Development/*` ist.

Folgen Sie den Schritten im vorherigen Abschnitt und aktualisieren Sie die Inline-Richtlinie noch einmal, die Sie für die Benutzerin Alice erstellt haben.

Kopieren Sie die folgende Richtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein, wobei die vorhandene Richtlinie ersetzt wird.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllowListBucketIfSpecificPrefixIsIncludedInRequest",
         "Action":["s3:ListBucket"],
         "Effect":"Allow",
         "Resource":["arn:aws:s3:::companybucket"],
         "Condition":{
            "StringLike":{"s3:prefix":["Development/*"]
            }
         }
      },
      {
        "Sid":"AllowUserToReadWriteObjectDataInDevelopmentFolder",
        "Action":["s3:GetObject", "s3:PutObject"],
        "Effect":"Allow",
        "Resource":["arn:aws:s3:::companybucket/Development/*"]
      },
      {
         "Sid": "ExplicitlyDenyAnyRequestsForAllOtherFoldersExceptDevelopment",
         "Action": ["s3:ListBucket"],
         "Effect": "Deny",
         "Resource": ["arn:aws:s3:::companybucket"],
         "Condition":{  "StringNotLike": {"s3:prefix":["Development/*",""] },
                        "Null"         : {"s3:prefix":false }
          }
      }
   ]
}
```

Schritt 6: Dem IAM-Benutzer Bob spezifische Berechtigungen erteilen

Sie können die Schritte wiederholen, um Bob eine ähnliche Berechtigung für den Ordner `Finance` zu erteilen. Führen Sie die Schritte aus, mit denen Sie zuvor Alice Berechtigungen erteilt haben, aber ersetzen Sie den Ordner `Development` durch den Ordner `Finance`. step-by-step Anweisungen finden Sie unter Schritt 5: Der IAM-Benutzerin Alice spezifische Berechtigungen erteilen.

Schritt 7: Absichern des Ordners „Private“

In diesem Beispiel haben Sie nur zwei Benutzer. Sie haben auf Gruppenebene alle erforderlichen Mindestberechtigungen erteilt und auf Benutzerebene die Berechtigungen nur dann gewährt, wenn die einzelnen Benutzer die Berechtigungen wirklich benötigen. Dieser Ansatz minimiert den Aufwand beim Verwalten der Berechtigungen. Wenn die Anzahl der Benutzer steigt, kann das Verwalten der Berechtigungen mühsam werden. Sie möchten z. B. nicht, dass irgendwelche der Benutzer in diesem Beispiel auf den Inhalt des Ordners `Private` zugreifen können. Wie stellen Sie sicher, dass Sie nicht versehentlich einem Benutzer Berechtigung darauf erteilen? Sie fügen eine Richtlinie hinzu, die explizit den Zugriff auf den Ordner verweigert. Eine explizite Zugriffsverweigerung überschreibt alle anderen Berechtigungen.

Um sicherzustellen, dass der Ordner `Private` auch privat bleibt, können Sie die folgenden beiden Ablehnungsanweisungen zur Gruppenrichtlinie hinzufügen:

- Fügen Sie die folgende Anweisung hinzu, um jede Aktion auf die Ressourcen im `Private`-Ordner (`companybucket/Private/*`) zu verweigern.

```
{
  "Sid": "ExplictDenyAccessToPrivateFolderToEveryoneInTheGroup",
  "Action": ["s3:*"],
  "Effect": "Deny",
  "Resource":["arn:aws:s3:::companybucket/Private/*"]
}
```

- Sie verweigern auch die Berechtigung für die Aktion Objekte auflisten, wenn die Anforderung das Präfix `Private/` angibt. Wenn Bob oder Alice in der Konsole den Order `Private` öffnen, bewirkt diese Richtlinie, dass Amazon S3 eine Fehlermeldung zurückgibt.

```
{
  "Sid": "DenyListBucketOnPrivateFolder",
  "Action": ["s3:ListBucket"],
  "Effect": "Deny",
  "Resource": ["arn:aws:s3:::*"],
  "Condition":{
      "StringLike":{"s3:prefix":["Private/"]}
   }
}
```

Ersetzen Sie die Gruppenrichtlinie `Consultants` durch eine aktualisierte Richtlinie, die die vorherigen Ablehnungsanweisungen enthält. Nachdem die aktualisierte Richtlinie angewendet wurde, kann keiner der Benutzer in der Gruppe mehr auf den Order `Private` in Ihrem Bucket zugreifen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   Verwenden Sie Ihre - AWS-Konto Anmeldeinformationen, nicht die eines IAM-Benutzers, um sich bei der Konsole anzumelden.

2. Ersetzen Sie die vorhandene von `AllowGroupToSeeBucketListInTheConsole` verwaltete Richtlinie, die der Gruppe `Consultants` zugeordnet ist, durch die folgende Richtlinie. Denken Sie daran, den Namen `companybucket` in der Richtlinie durch den Namen Ihres Buckets zu ersetzen.

   Weitere Anweisungen finden Sie unter Bearbeiten von Kunden verwalteter Richtlinien im IAM-Benutzerhandbuch. Wenn Sie die Anweisungen nachvollziehen, beachten Sie bitte die Anweisungen zum Ändern aller Hauptentitäten, denen die Richtlinie zugeordnet ist.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":
 "AllowGroupToSeeBucketListAndAlsoAllowGetBucketLocationRequiredForListBucket",
      "Action": ["s3:ListAllMyBuckets", "s3:GetBucketLocation"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::*"]
    },
    {
      "Sid": "AllowRootLevelListingOfCompanyBucket",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::companybucket"],
      "Condition":{
          "StringEquals":{"s3:prefix":[""]}
       }
    },
    {
      "Sid": "RequireFolderStyleList",
      "Action": ["s3:ListBucket"],
      "Effect": "Deny",
```

```
      "Resource": ["arn:aws:s3:::*"],
      "Condition":{
          "StringNotEquals":{"s3:delimiter":"/"}
       }
     },
    {
      "Sid": "ExplictDenyAccessToPrivateFolderToEveryoneInTheGroup",
      "Action": ["s3:*"],
      "Effect": "Deny",
      "Resource":["arn:aws:s3:::companybucket/Private/*"]
    },
    {
      "Sid": "DenyListBucketOnPrivateFolder",
      "Action": ["s3:ListBucket"],
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::*"],
      "Condition":{
          "StringLike":{"s3:prefix":["Private/"]}
       }
    }
  ]
}
```

Schritt 8: Bereinigen

Öffnen Sie zum Bereinigen die IAM-Konsole und entfernen Sie die Benutzer Alice und Bob. step-by-step Anweisungen finden Sie unter Löschen eines IAM-Benutzers im IAM-Benutzerhandbuch.

Um sicherzustellen, dass Sie für die Speicherung nicht künftig belastet werden, sollten Sie auch die Objekte und den Bucket löschen, die Sie für diese Übung erstellt haben.

Zugehörige Ressourcen

- Verwalten von IAM-Richtlinien im IAM-Benutzerhandbuch.

Beispiele für Benutzer- und Rollenrichtlinien

Dieser Abschnitt zeigt mehrere Beispielrichtlinien für AWS Identity and Access Management (IAM)-Benutzer und -Rollen zur Steuerung des Zugriffs auf Amazon S3. Beispiele für Bucket-Richtlinien finden Sie unter Verwenden von Bucket-Richtlinien. Informationen zur IAM-Richtliniensprache finden Sie unter Bucket-Richtlinien und Benutzerrichtlinien.

Die folgenden Beispielrichtlinien funktionieren, wenn Sie sie programmgesteuert verwenden. Um sie mit der Amazon-S3-Konsole verwenden zu können, müssen Sie aber zusätzliche Berechtigungen gewähren, die für die Konsole erforderlich sind. Informationen zur Verwendung von Richtlinien wie dieser mit der Amazon-S3-Konsole finden Sie unter Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien.

Themen

Einem IAM-Benutzer den Zugriff auf einen Ihrer Buckets erlauben

In diesem Beispiel möchten Sie einem IAM-Benutzer in Ihrem AWS-Konto Zugriff auf einen Ihrer Buckets gewähren, *DOC-EXAMPLE-BUCKET1*, und dem Benutzer erlauben, Objekte hinzuzufügen, zu aktualisieren und zu löschen.

Zusätzlich zum Erteilen der Berechtigungen `s3:PutObject`, `s3:GetObject` und `s3:DeleteObject` für den Benutzer, gewährt die Richtlinie die Berechtigungen `s3:ListAllMyBuckets`, `s3:GetBucketLocation` und `s3:ListBucket`. Dies sind die zusätzlichen Berechtigungen, die von der Konsole benötigt werden. Außerdem sind die Aktionen `s3:PutObjectAcl` und `s3:GetObjectAcl` erforderlich, um Objekte in der Konsole kopieren, ausschneiden und einfügen zu können. Ein detailliertes Beispiel für eine Richtlinie, die Berechtigungen für Benutzer erteilt und sie unter Verwendung der Konsole testet, finden Sie unter Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
         "Effect":"Allow",
         "Action": "s3:ListAllMyBuckets",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":["s3:ListBucket","s3:GetBucketLocation"],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:GetObject",
            "s3:GetObjectAcl",
            "s3:DeleteObject"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
      }
   ]
}
```

Jedem IAM-Benutzer Zugriff auf einen Ordner in einem Bucket erlauben

In diesem Beispiel möchten Sie zwei IAM-Benutzern, Mary und Carlos, den Zugriff auf Ihren Bucket, *DOC-EXAMPLE-BUCKET1*, gewähren, damit sie Objekte hinzufügen, aktualisieren und löschen können. Allerdings möchten Sie den Zugriff beider Benutzer auf ein einzelnes Präfix (Ordner) im Bucket einschränken. Sie könnten Ordner mit Namen erstellen, die dem jeweiligen Benutzernamen entsprechen.

```
DOC-EXAMPLE-BUCKET1
   Mary/
   Carlos/
```

Um jedem Benutzer nur den Zugriff auf den eigenen Ordner zu gewähren, können Sie für jeden Benutzer eine Richtlinie schreiben und ihnen einzeln zuweisen. Sie können beispielsweise die folgende Richtlinie der Benutzerin Mary zuweisen, um ihr spezifische Amazon-S3-Berechtigungen für den Ordner *DOC-EXAMPLE-BUCKET1*/*Mary* zu gewähren.

```
{
```

```
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/Mary/*"
      }
   ]
}
```

Dann können Sie dem Benutzer Carlos eine ähnliche Richtlinie zuweisen, indem Sie den Ordner *Carlos*im Wert `Resource` angeben.

Statt Richtlinien einzelnen Benutzern zuzuweisen, können Sie auch eine einzelne Richtlinie mit einer Richtlinienvariable schreiben und dann die Richtlinie einer Gruppe zuweisen. Sie müssen zuerst eine Gruppe erstellen und die Benutzer Mary und Carlos in die Gruppe aufnehmen. Die folgende Beispielrichtlinie erlaubt eine Reihe von Amazon-S3-Berechtigungen für den Ordner *DOC-EXAMPLE-BUCKET1*`/${aws:username}`. Wenn die Richtlinie ausgewertet wird, wird die Richtlinienvariable `${aws:username}` durch den Benutzernamen des Anforderers ersetzt. Wenn Mary beispielsweise eine Anforderung zum Anlegen eines Objekts sendet, ist die Operation nur zulässig, wenn Mary das Objekt in den Ordner *DOC-EXAMPLE-BUCKET1*`/Mary` hochlädt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/${aws:username}/*"
      }
```

```
   ]
}
```

> **Note**
>
> Bei der Verwendung von Richtlinienvariablen müssen Sie explizit die Version `2012-10-17` in der Richtlinie angeben. Die Standardversion der IAM-Richtliniensprache, 2008-10-17, unterstützt keine Richtlinienvariablen.

Wenn Sie die vorherige Richtlinie in der Amazon-S3-Konsole testen möchten, erfordert die Konsole zusätzliche Berechtigungen, wie in der folgenden Richtlinie gezeigt. Weitere Informationen darüber, wie die Konsole diese Berechtigungen verwendet, finden Sie unter Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien.

```
{
 "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGroupToSeeBucketListInTheConsole",
      "Action": [
       "s3:ListAllMyBuckets",
       "s3:GetBucketLocation"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Sid": "AllowRootLevelListingOfTheBucket",
      "Action": "s3:ListBucket",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
      "Condition":{
            "StringEquals":{
                    "s3:prefix":[""], "s3:delimiter":["/"]
                           }
                }
    },
    {
      "Sid": "AllowListBucketOfASpecificUserPrefix",
      "Action": "s3:ListBucket",
      "Effect": "Allow",
```

```
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
      "Condition":{  "StringLike":{"s3:prefix":["${aws:username}/*"] }
       }
    },
      {
     "Sid": "AllowUserSpecificActionsOnlyInTheSpecificUserPrefix",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/${aws:username}/*"
      }
  ]
}
```

> **Note**
>
> In der 2012-10-17-Version der Richtlinie beginnen Richtlinienvariablen mit $. Diese Syntaxänderung kann möglicherweise einen Konflikt verursachen, wenn Ihr Objektschlüssel (Objektname) ein $-Zeichen enthält.
> Damit dieser Konflikt vermieden wird, geben Sie das Zeichen $ mithilfe von `${$}` an. Um beispielsweise einen Objektschlüssel `my$file` in eine Richtlinie aufzunehmen, geben Sie das Zeichen mit `my${$}file` an.

Obwohl IAM-Benutzernamen benutzerfreundliche, von Menschen lesbare Bezeichner sind, müssen sie global nicht eindeutig sein. Wenn beispielsweise der Benutzer Carlos die Organisation verlässt und ein anderer Carlos hinzukommt, könnte der neue Carlos auf die Informationen des vorherigen Carlos zugreifen.

Anstatt Benutzernamen zu verwenden, könnten Sie Ordner erstellen, die auf IAM-Benutzer-IDs basieren. Jede IAM-Benutzer-ID muss eindeutig sein. In diesem Fall müssen Sie die vorherige Richtlinie ändern, und die Richtlinienvariable `${aws:userid}` verwenden. Weitere Informationen zu den Benutzerkennungen finden Sie unter IAM-Kennungen im IAM-Benutzerhandbuch.

```
{
```

```
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/home/${aws:userid}/*"
      }
   ]
}
```

Nicht-IAM-Benutzern (Nutzern mobiler Apps) den Zugriff auf Ordner in einem Bucket erlauben

Angenommen, Sie möchten eine mobile App entwickeln, ein Spiel, in dem Benutzerdaten in einem S3-Bucket gespeichert werden. Sie möchten für jeden App-Benutzer einen Ordner in Ihrem Bucket erstellen. Sie möchten auch den Zugriff jedes Benutzers auf seinen eigenen Ordner beschränken. Sie können jedoch keine Ordner erstellen, bevor jemand Ihre App herunterlädt und das Spiel beginnt, weil Sie dessen Benutzer-ID nicht vorliegen haben.

In diesem Fall können Sie von den Nutzern verlangen, dass Sie sich in Ihrer App über einen öffentlichen Identitätsanbieter anmelden, wie z. B. Login with Amazon, Facebook oder Google. Nachdem sich Benutzer über einen dieser Anbieter bei Ihrer App angemeldet haben, verfügen sie über eine Benutzer-ID, mit der Sie zur Laufzeit benutzerspezifische Ordner erstellen können.

Anschließend können Sie den Web-Identitätsverbund in verwenden AWS Security Token Service , um Informationen vom Identitätsanbieter in Ihre App zu integrieren und temporäre Sicherheitsanmeldeinformationen für jeden Benutzer zu erhalten. Sie können dann IAM-Richtlinien erstellen, die es der App ermöglichen, auf Ihren Bucket zuzugreifen und solche Vorgänge wie das Erstellen von benutzerspezifischen Ordnern und das Hochladen von Daten durchzuführen. Weitere Informationen zum Web-Identitätsverbund finden Sie unter Über Web Identity Federation im IAM-Benutzerhandbuch.

Einer Gruppe erlauben, einen freigegebenen Ordner in Amazon S3 zu haben

Durch Anfügen der folgenden Richtlinie an die Gruppe erhalten alle Benutzer der Gruppe Zugriff auf den folgenden Ordner in Amazon S3: *DOC-EXAMPLE-BUCKET1*/share/marketing.

Gruppenmitglieder dürfen nur auf die spezifischen Amazon-S3-Berechtigungen zugreifen, die in der Richtlinie gegeben sind, und nur für Objekte im angegebenen Ordner.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/share/marketing/*"
      }
   ]
}
```

Erteilen der Erlaubnis für alle Benutzer, Objekte in einem Teil des Buckets zu lesen

In diesem Beispiel erstellen Sie eine Gruppe mit dem Namen *AllUsers*, die alle IAM-Benutzer enthält, die dem AWS-Konto angehören. Anschließend fügen Sie eine Richtlinie hinzu, die der Gruppe Zugriff auf `GetObject` und `GetObjectVersion` gewährt, jedoch nur für Objekte im Ordner *DOC-EXAMPLE-BUCKET1*`/readonly`.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/readonly/*"
      }
   ]
}
```

Erteilen der Erlaubnis für einen Partner, Dateien in einem bestimmten Bereich des Buckets abzulegen

In diesem Beispiel erstellen Sie eine Gruppe namens *AnyCompany*, die eine Partnerfirma darstellt. Sie erstellen einen IAM-Benutzer für die bestimmte Person oder Anwendung bei der Partnerfirma, die Zugriff benötigt, und dann fügen Sie den Benutzer in die Gruppe ein.

Sie fügen dann eine Richtlinie hinzu, die der Gruppe den `PutObject`-Zugriff auf den folgenden Ordner im Bucket erteilt:

DOC-EXAMPLE-BUCKET1`/uploads/anycompany`

Sie möchten verhindern, dass die *AnyCompany*-Gruppe andere Aktionen für den Bucket ausführt. Daher fügen Sie eine Anweisung hinzu, die explizit die Berechtigung für andere Amazon-S3-Aktionen verweigert, außer für `PutObject` in einer Amazon-S3-Ressource im AWS-Konto.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"s3:PutObject",
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/uploads/anycompany/*"
      },
      {
         "Effect":"Deny",
         "Action":"s3:*",
         "NotResource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/uploads/anycompany/*"
      }
   ]
}
```

Beschränken des Zugriffs auf Amazon-S3-Buckets in einem bestimmten AWS-Konto

Wenn Sie sicherstellen möchten, dass Ihre Amazon S3-Prinzipale nur auf die Ressourcen zugreifen, die sich innerhalb eines vertrauenswürdigen befinden AWS-Konto, können Sie den Zugriff einschränken. Beispiel: Diese identitätsbasierte IAM-Richtlinie verwendet eine `Deny`-Auswirkung, um den Zugriff auf Amazon-S3-Aktionen zu blockieren, es sei denn, die Amazon-S3-Ressource, auf die zugegriffen wird, befindet sich im Konto *222222222222*. Um zu verhindern, dass ein IAM AWS-Konto -Prinzipal in einem auf Amazon S3-Objekte außerhalb des Kontos zugreift, fügen Sie die folgende IAM-Richtlinie an:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyS3AccessOutsideMyBoundary",
      "Effect": "Deny",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "222222222222"
          ]
        }
      }
    }
  ]
}
```

> **Note**
>
> Diese Richtlinie ersetzt Ihre vorhandenen IAM-Zugriffskontrollen nicht, da sie keinen Zugriff gewährt. Stattdessen fungiert diese Richtlinie als zusätzlicher Integritätsschutz für Ihre anderen IAM-Berechtigungen, unabhängig von den durch andere IAM-Richtlinien erteilten Berechtigungen.

Ersetzen Sie unbedingt die Konto-ID *222222222222* in der Richtlinie durch Ihr eigenes AWS-Konto. Wenn Sie eine Richtlinie unter Beibehaltung dieser Einschränkung auf mehrere Konten anwenden möchten, ersetzen Sie die Konto-ID durch den Bedingungsschlüssel `aws:PrincipalAccount`. Diese Bedingung erfordert, dass sich der Prinzipal und die Ressource in demselben Konto befinden müssen.

Beschränken des Zugriffs auf Amazon-S3-Buckets innerhalb Ihrer Organisationseinheit (OU)

Wenn Sie eine Organisationseinheit (OU) in eingerichtet haben AWS Organizations, sollten Sie den Amazon S3-Bucket-Zugriff auf einen bestimmten Teil Ihrer Organisation beschränken. In diesem Beispiel wird der `aws:ResourceOrgPaths`-Schlüssel verwendet, um den Amazon-S3-Bucket-

Zugriff auf eine OU in Ihrer Organisation einzuschränken. In diesem Beispiel lautet die OU-ID *ou-acroot-exampleou*. Stellen Sie sicher, dass Sie diesen Wert in Ihrer Richtlinie durch Ihre eigenen OU-IDs ersetzen.

```
{
   "Version": "2012-10-17",
   "Statement": [
     {
       "Sid": "AllowS3AccessOutsideMyBoundary",
       "Effect": "Allow",
       "Action": [
         "s3:*"
       ],
       "Resource": "*",
       "Condition": {
         "ForAllValues:StringNotLike": {
           "aws:ResourceOrgPaths": [
             "o-acorg/r-acroot/ou-acroot-exampleou/"
           ]
         }
       }
     }
   ]
 }
```

> **Note**
>
> Diese Richtlinie gewährt keinen Zugriff. Stattdessen fungiert diese Richtlinie als Schutz für Ihre anderen IAM-Berechtigungen und verhindert, dass Ihre Prinzipale außerhalb einer von OU definierten Grenze auf Amazon-S3-Objekte zugreifen.

Die Richtlinie verweigert den Zugriff auf Amazon-S3-Aktionen, es sei denn, das Amazon-S3-Objekt, auf das zugegriffen wird, befindet sich in der OU *ou-acroot-exampleou* Ihrer Organisation. Die IAM-Richtlinienbedingung erfordert, dass `aws:ResourceOrgPaths`, ein Mehrfachwertbedingungsschlüssel, einen der aufgelisteten OU-Pfade enthält. Die Richtlinie verwendet den Operator `ForAllValues:StringNotLike` zum Vergleich der Werte von `aws:ResourceOrgPaths` mit den aufgelisteten OUs ohne Berücksichtigung der Groß- und Kleinschreibung.

Beschränken des Zugriffs auf Amazon-S3-Buckets innerhalb Ihrer Organisation

Wenn Sie den Zugriff auf Amazon-S3-Objekte in Ihrer Organisation einschränken möchten, fügen Sie eine IAM-Richtlinie an das Stammverzeichnis der Organisation an und wenden Sie sie auf alle Konten in Ihrer Organisation an. Verwenden Sie eine Service-Kontrollrichtlinie (SCP), um zu verlangen, dass Ihre IAM-Prinzipale diese Regel befolgen. Wenn Sie sich für die Verwendung einer SCP entscheiden, achten Sie darauf, die SCP gründlich zu testen, bevor Sie die Richtlinie dem Stammverzeichnis der Organisation anfügen.

In der folgenden Beispielrichtlinie wird der Zugriff auf Amazon-S3-Aktionen verweigert, es sei denn, das Amazon-S3-Objekt, auf das zugegriffen wird, befindet sich in derselben Organisation wie der IAM-Prinzipal, der darauf zugreift:

```
{
   "Version": "2012-10-17",
   "Statement": [
     {
       "Sid": "DenyS3AccessOutsideMyBoundary",
       "Effect": "Deny",
       "Action": [
         "s3:*"
       ],
       "Resource": "arn:aws:s3:::*/*",
       "Condition": {
         "StringNotEquals": {
           "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
         }
       }
     }
   ]
 }
```

> **Note**
>
> Diese Richtlinie gewährt keinen Zugriff. Stattdessen fungiert diese Richtlinie als Schutz für Ihre anderen IAM-Berechtigungen und verhindert, dass Ihre Prinzipale auf Amazon-S3-Objekte außerhalb Ihrer Organisation zugreifen. Diese Richtlinie gilt auch für Amazon-S3-Ressourcen, die nach Inkrafttreten der Richtlinie erstellt werden.

Die IAM-Richtlinienbedingung in diesem Beispiel erfordert, dass `aws:ResourceOrgID` und `aws:PrincipalOrgID` gleich sind. Bei dieser Anforderung müssen sich der Prinzipal, der die Anforderung stellt, und die Ressource, auf die zugegriffen wird, in derselben Organisation befinden.

Beispiel-Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon-S3-Ressourcen

In diesem Thema werden die folgenden einführenden Anleitungsbeispiele für das Gewähren des Zugriffs aufs Amazon-S3-Ressourcen bereitgestellt. Diese Beispiele verwenden die , AWS Management Console um Ressourcen (Buckets, Objekte, Benutzer) zu erstellen und ihnen Berechtigungen zu erteilen. Anschließend zeigen die Beispiele auf, wie Sie Berechtigungen mithilfe der Befehlszeilen-Tools überprüfen können, sodass Sie keinen Code schreiben müssen. Wir stellen Befehle bereit, die sowohl die AWS Command Line Interface (CLI) als auch die verwenden AWS Tools for Windows PowerShell.

- Beispiel 1: Bucket-Eigentümer erteilt seinen Benutzern Bucket-Berechtigungen

  Die in Ihrem Konto erstellten IAM-Benutzer verfügen standardmäßig über keine Berechtigungen. In dieser Übung erteilen Sie einem Benutzer die Berechtigung, Bucket- und Objekt-Vorgänge auszuführen.

- Beispiel 2: Bucket-Eigentümer erteilt kontoübergreifende Bucket-Berechtigungen

  In dieser Übung gewährt ein Bucket-Eigentümer, Konto A, einem anderen AWS-Konto, Konto B, kontoübergreifende Berechtigungen. Konto B delegiert diese Berechtigungen anschließend an Benutzer in seinem Konto.

- Verwalten von Objektberechtigungen, wenn der Objekt- und der Bucket-Eigentümer nicht identisch sind

  Bei den Beispielszenarien in diesem Fall geht es um einen Bucket-Eigentümer, der anderen Objektberechtigungen erteilt, es gehören jedoch nicht alle Objekte im Bucket dem Bucket-Eigentümer. Welche Berechtigungen benötigt der Bucket-Eigentümer und wie kann er diese Berechtigungen delegieren?

  Das AWS-Konto , das einen Bucket erstellt, wird als Bucket-Eigentümer bezeichnet. Der Eigentümer kann anderen die AWS-Konten Berechtigung zum Hochladen von Objekten erteilen, und die AWS-Konten , die Objekte erstellen, besitzen diese. Der Bucket-Eigentümer hat keine Berechtigungen für diese Objekte, die von anderen AWS-Konten erstellt wurden. Wenn der Bucket-Eigentümer eine Bucket-Richtlinie erstellt, die Zugriff auf Objekte erteilt, gilt diese Richtlinie nicht für Objekte, die sich im Besitz von anderen Konten befinden.

  In diesem Fall muss der Objekteigentümer zuerst dem Bucket-Eigentümer über eine Objekt-ACL Berechtigungen erteilen. Der Bucket-Eigentümer kann diese Objektberechtigungen dann an andere

delegieren, an Benutzer in seinem eigenen Konto oder an ein anderes , AWS-Konto wie in den folgenden Beispielen dargestellt.

- Beispiel 3: Bucket-Eigentümer, der Berechtigungen für Objekte erteilt, die ihm nicht gehören

  In dieser Übung erhält der Bucket-Eigentümer zuerst Berechtigungen von dem Objekteigentümer. Der Bucket-Eigentümer delegiert diese Berechtigungen anschließend an Benutzer in seinem eigenen Konto.

- Beispiel 4: Der Bucket-Eigentümer erteilt eine kontenübergreifende Berechtigung für Objekte, die ihm nicht gehören

  Nach Erhalt von Berechtigungen vom Objekteigentümer kann der Bucket-Eigentümer die Berechtigung nicht an andere delegieren, AWS-Konten da die kontoübergreifende Delegierung nicht unterstützt wird (siehe Berechtigungsdelegation). Stattdessen kann der Bucket-Eigentümer eine IAM-Rolle mit Berechtigungen erstellen, um bestimmte Operationen auszuführen (z. B. Objekte abrufen) und einem anderen erlauben, diese Rolle AWS-Konto zu übernehmen. Jeder, der diese Rolle annimmt, kann anschließend auf Objekte zugreifen. Dieses Beispiel zeigt, wie ein Bucket-Eigentümer diese kontoübergreifende Delegation mithilfe einer IAM-Rolle aktivieren kann.

Bevor Sie die beispiehalften Walkthroughs ausprobieren

In diesen Beispielen wird die verwendet AWS Management Console , um Ressourcen zu erstellen und Berechtigungen zu erteilen. Und um Berechtigungen zu testen, verwenden die Beispiele die Befehlszeilen-Tools, AWS Command Line Interface (CLI) und AWS Tools for Windows PowerShell, sodass Sie keinen Code schreiben müssen. Zum Testen der Berechtigungen müssen Sie eins dieser Tools einrichten. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

Darüber hinaus verwenden diese Beispiele beim Erstellen von Ressourcen keine Root-Benutzer-Anmeldeinformationen von einem AWS-Konto. Sie erstellen stattdessen einen Administratorbenutzer in diesen Konten, um diese Aufgaben auszuführen.

Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen

AWS Identity and Access Management (IAM) rät davon ab, die Root-Benutzer-Anmeldeinformationen Ihres für Anfragen AWS-Konto zu verwenden. Erstellen Sie stattdessen eine(n) IAM-Benutzer oder eine -Rolle, gewähren Sie diesem/r vollständigen Zugriff und verwenden Sie anschließend die Anmeldeinformationen dieses Benutzers/dieser Rolle zum Erstellen von Anfragen. Wir bezeichnen

dies als Administratorbenutzer oder -rolle. Weitere Informationen finden Sie unter Root-Benutzer des AWS-Kontos -Anmeldeinformationen und IAM-Identitäten in der Allgemeine AWS-Referenz und unter Bewährte IAM-Methoden im IAM-Benutzerhandbuch.

Alle Beispielanleitungen in diesem Abschnitt verwenden die Anmeldeinformationen des Administratorbenutzers. Wenn Sie keinen Administratorbenutzer für Ihr erstellt haben AWS-Konto, zeigen Ihnen die Themen die Vorgehensweise.

Beachten Sie, dass Sie die URL für die Anmeldung von IAM-Benutzern verwenden müssen, um sich AWS Management Console mit den Benutzeranmeldeinformationen bei der anzumelden. Die IAM-Konsole stellt diese URL für Ihr bereit AWS-Konto. In diesen Themen erfahren Sie, wie Sie die URL abrufen können.

Einrichten der Tools für die beispielhaften Walkthroughs

In den einführenden Beispielen (siehe Beispiel-Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon-S3-Ressourcen) wird die verwendet AWS Management Console , um Ressourcen zu erstellen und Berechtigungen zu erteilen. Und um Berechtigungen zu testen, verwenden die Beispiele die Befehlszeilen-Tools AWS Command Line Interface (CLI) und AWS Tools for Windows PowerShell, sodass Sie keinen Code schreiben müssen. Zum Testen der Berechtigungen müssen Sie eins dieser Tools einrichten.

So richten Sie die ein AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch.

   Einrichtung der AWS Command Line Interface

   Installieren der AWS Command Line Interface

   Konfigurieren der AWS Command Line Interface
2. Richten Sie das Standardprofil ein.

   Sie speichern Benutzeranmeldeinformationen in der AWS CLI Konfigurationsdatei. Erstellen Sie mit Ihren - AWS-Konto Anmeldeinformationen ein Standardprofil in der Konfigurationsdatei. Anweisungen zum Suchen und Bearbeiten Ihrer Konfigurationsdatei finden Sie unter Konfigurations- und Anmeldeinformationsdateien. AWS CLI

   ```
   [default]
   ```

```
aws_access_key_id = access key ID
aws_secret_access_key = secret access key
region = us-west-2
```

3. Überprüfen Sie die Einrichtung, indem Sie den folgenden Befehl in die Befehlszeile eingeben. Beide Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Probieren Sie den Hilfebefehl aus.

     ```
     aws help
     ```

   - Verwenden Sie `aws s3 ls`, um eine Liste der Buckets auf dem konfigurierten Konto abzurufen.

     ```
     aws s3 ls
     ```

Im Verlauf dieser Beispiel-Anleitungen erstellen Sie Benutzer und speichern Anmeldeinformationen in den Config-Dateien, indem Sie Profile erstellen. Dies wird im folgenden Beispiel dargestellt. Beachten Sie, dass diese Profile Namen haben (AccountAadmin und AccountBadmin):

```
[profile AccountAadmin]
aws_access_key_id = User AccountAadmin access key ID
aws_secret_access_key = User AccountAadmin secret access key
region = us-west-2

[profile AccountBadmin]
aws_access_key_id = Account B access key ID
aws_secret_access_key = Account B secret access key
region = us-east-1
```

Um mit diesen Benutzeranmeldeinformationen einen Befehl auszuführen, fügen Sie den Parameter `--profile` hinzu, um den Profilnamen festzulegen. Der folgende AWS CLI Befehl ruft eine Liste von Objekten in ab `examplebucket` und gibt das `AccountBadmin` Profil an.

```
aws s3 ls s3://examplebucket --profile AccountBadmin
```

Alternativ können Sie eine Reihe von Anmeldeinformationen als Standardprofil konfigurieren, indem Sie die Umgebungsvariable `AWS_DEFAULT_PROFILE` von der Befehlszeile aus ändern.

Wenn Sie dies getan haben, verwendet die jedes Mal, wenn Sie AWS CLI Befehle ohne den `--profile`Parameter ausführen, das Profil, das Sie in der Umgebungsvariablen als Standardprofil festgelegt AWS CLI haben.

```
$ export AWS_DEFAULT_PROFILE=AccountAadmin
```

So richten Sie ein AWS Tools for Windows PowerShell

1. Herunterladen und Konfigurieren von AWS Tools for Windows PowerShell. Anweisungen finden Sie unter Download and Install the AWS Tools for Windows PowerShell(Herunterladen und Installieren der TWP) im AWS Tools for Windows PowerShell -Benutzerhandbuch.

   > **Note**
   >
   > Um das AWS Tools for Windows PowerShell Modul zu laden, müssen Sie die PowerShell Skriptausführung aktivieren. Weitere Informationen finden Sie unter Enable Script Execution (Skript-Ausführung aktivieren) im AWS Tools for Windows PowerShell -Benutzerhandbuch.

2. Für diese Übungen geben Sie mit dem `Set-AWSCredentials` Befehl AWS Anmeldeinformationen pro Sitzung an. Der Befehl speichert die Anmeldeinformationen in einem persistenten Speicher (Parameter `-StoreAs`).

   ```
   Set-AWSCredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -
   storeas string
   ```

3. Überprüfen Sie die Einrichtung.

   - Führen Sie `Get-Command` aus, um eine Liste der verfügbaren Befehle für Amazon-S3-Vorgänge abzurufen.

     ```
     Get-Command -module awspowershell -noun s3* -StoredCredentials string
     ```

   - Führen Sie den Befehl `Get-S3Object` aus, um eine Liste von Objekten in einem Bucket abzurufen.

     ```
     Get-S3Object -BucketName bucketname -StoredCredentials string
     ```

Eine Liste mit Befehlen finden Sie unter Amazon Simple Storage Service-Cmdlets.

Jetzt können Sie die Übungen ausprobieren. Folgen Sie den Links am Anfang des Abschnitts.

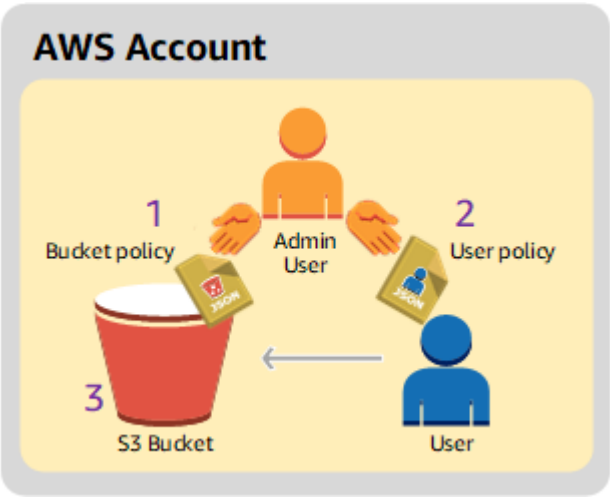
Beispiel 1: Bucket-Eigentümer erteilt seinen Benutzern Bucket-Berechtigungen

> **Important**
>
> Das Erteilen von Berechtigungen für IAM-Rollen ist eine bessere Vorgehensweise als die Erteilung von Berechtigungen an einzelne Benutzer. Weitere Informationen zur Vorgehensweise finden Sie unter Hintergrund: Kontoübergreifende Berechtigungen und die Verwendung von IAM-Rollen.

Themen

In dieser Übung AWS-Konto besitzt ein einen Bucket und es hat einen IAM-Benutzer im Konto. Standardmäßig hat der Benutzer keine Berechtigungen. Damit der Benutzer alle Aufgaben ausführen kann, muss das übergeordnete Konto ihm Berechtigungen erteilen. Der Bucket-Eigentümer und das übergeordnete Konto sind identisch. Um den Benutzern Berechtigungen für den Bucket zu erteilen, AWS-Konto kann die daher eine Bucket-Richtlinie, eine Benutzerrichtlinie oder beides verwenden. Der Kontobesitzer gewährt einige Berechtigungen unter Verwendung einer Bucket-Richtlinie und andere Berechtigungen unter Verwendung einer Benutzerrichtlinie.

Die folgenden Schritte fasen das detaillierte Beispiel zusammen:



1. Der Kontoadministrator erstellt eine Bucket-Richtlinie, die dem Benutzer verschiedene Berechtigungen erteilt.
2. Der Kontoadministrator weist dem Benutzer eine Benutzerrichtlinie zu, die ihm zusätzliche Berechtigungen erteilt.
3. Anschließend probiert der Benutzer Berechtigungen aus, die über die Bucket-Richtlinie und die Benutzerrichtlinie erteilt wurden.

Für dieses Beispiel benötigen Sie eine AWS-Konto. Anstatt die Anmeldeinformationen des Root-Benutzers für das Konto zu verwenden, erstellen Sie einen Administrator-Benutzer (siehe Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen). Wir verweisen wie folgt auf die AWS-Konto und den Administratorbenutzer:

Konto-ID	Konto bezeichnet als	Administratorbenutzer im Konto
1111-1111-1111	Konto A	AccountAadmin

> **Note**
>
> Der Administratorbenutzer in diesem Beispiel ist AccountAadmin, was sich auf Konto A bezieht, und nicht AccountAdmin.

Alle Aufgaben in Verbindung mit dem Erstellen von Benutzern und Gewähren von Berechtigungen werden in der AWS Management Console ausgeführt. Um die Berechtigungen zu überprüfen, verwendet die Anleitung die Befehlszeilen-Tools (AWS Command Line Interface CLI) und AWS Tools for Windows PowerShell, um die Berechtigungen zu überprüfen, sodass Sie keinen Code schreiben müssen.

Schritt 0: Vorbereitung auf den Walkthrough

1. Stellen Sie sicher, dass Sie über ein verfügen AWS-Konto und dass es einen Benutzer mit Administratorrechten hat.

   a. Melden Sie sich nach Bedarf für ein -Konto an. Wir bezeichnen dieses Konto als Konto A.

i. Rufen Sie https://aws.amazon.com/s3 auf und klicken Sie auf Sign Up (Registrieren).

ii. Folgen Sie den Anweisungen auf dem Bildschirm.

AWS benachrichtigt Sie per E-Mail, wenn Ihr Konto aktiv ist und Ihnen zur Nutzung zur Verfügung steht.

b. Erstellen Sie in Konto A einen Administratorbenutzer AccountAadmin. Melden Sie sich mit den Anmeldeinformationen von Konto A in der IAM-Konsole an und gehen Sie wie folgt vor:

i. Erstellen Sie einen Benutzer AccountAadmin und notieren Sie sich die Sicherheitsanmeldeinformationen des Benutzers.

Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.

ii. Erteilen Sie AccountAadmin Administratorrechte, indem Sie eine Benutzerrichtlinie anfügen, die vollen Zugriff gewährt.

Weitere Informationen finden Sie unter Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.

iii. Notieren Sie sich die Anmelde-URL des IAM-Benutzers für AccountAadmin. Sie brauchen diese URL, wenn Sie sich bei der AWS Management Console anmelden. Weitere Informationen darüber, wo Sie diese finden, finden Sie unter Wie sich Benutzer in Ihrem Konto anmelden im IAM-Benutzerhandbuch. Notieren Sie die URLs für alle Konten.

2. Richten Sie entweder die AWS Command Line Interface (CLI) oder die ein AWS Tools for Windows PowerShell. Stellen Sie sicher, dass Sie die Anmeldeinformationen speichern, wie folgt:

- Wenn Sie die verwenden AWS CLI, erstellen Sie ein Profil AccountAadminin der Konfigurationsdatei.
- Wenn Sie die verwenden AWS Tools for Windows PowerShell, stellen Sie sicher, dass Sie die Anmeldeinformationen für die Sitzung als speichern AccountAadmin.

Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

Schritt 1: Erstellen von Ressourcen (einen Bucket und einen IAM-Benutzer) in Konto A und Erteilen von Berechtigungen

Melden Sie sich mit den Anmeldeinformationen des Benutzers AccountAadmin in Konto A und der speziellen Anmelde-URL des IAM-Benutzers bei der an AWS Management Console und gehen Sie wie folgt vor:

1. Ressourcen erstellen (einen Bucket und einen IAM-Benutzer)

   a. Erstellen Sie in der Amazon-S3-Konsole einen Bucket. Notieren Sie sich die , AWS-Region in der Sie sie erstellt haben. Anweisungen finden Sie unter Erstellen eines Buckets.

   b. Gehen Sie in der IAM-Konsole wie folgt vor:

      i. Erstellen Sie einen Benutzer, Dave.

         step-by-step Anweisungen finden Sie unter Erstellen von IAM-Benutzern (AWS Management Console) im IAM-Benutzerhandbuch.

      ii. Notieren Sie sich die UserDave Anmeldeinformationen.

      iii. Notieren Sie den Amazon-Ressourcennamen (ARN) für den Benutzer Dave. Wählen Sie in der IAM-Konsole den Benutzer aus, und auf der Registerkarte Zusammenfassung wird der Benutzer-ARN angezeigt.

2. Erteilen Sie Berechtigungen.

   Da der Bucket-Eigentümer und das übergeordnete Konto, zu dem der Benutzer gehört, identisch sind, AWS-Konto kann die Benutzerberechtigungen mithilfe einer Bucket-Richtlinie, einer Benutzerrichtlinie oder beidem erteilen. In diesem Beispiel machen Sie beides. Wenn das Objekt auch demselben Konto gehört, kann der Bucket-Eigentümer in der Bucket-Richtlinie (oder einer IAM-Richtlinie) Objektberechtigungen erteilen.

   a. Fügen Sie in der Amazon-S3-Konsole die folgende Bucket-Richtlinie an *awsexamplebucket1* an.

      Die Richtlinie enthält zwei Anweisungen.

      - Die erste Anweisung erteilt Dave Berechtigungen für die Bucket-Operationen `s3:GetBucketLocation` und `s3:ListBucket`.
      - Die zweite Anweisung erteilt die `s3:GetObject`-Berechtigung. Konto A gehört auch das Objekt, deshalb kann der Kontoadministrator die `s3:GetObject`-Berechtigung erteilen.

In der `Principal`-Anweisung wird Dave durch seinen Benutzer-ARN identifiziert. Weitere Informationen zu Richtlinienelementen finden Sie unter Bucket-Richtlinien und Benutzerrichtlinien.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "statement1",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
            "s3:GetBucketLocation",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::awsexamplebucket1"
         ]
      },
      {
         "Sid": "statement2",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
             "s3:GetObject"
         ],
         "Resource": [
            "arn:aws:s3:::awsexamplebucket1/*"
         ]
      }
   ]
}
```

b. Erstellen Sie mithilfe der folgenden Richtlinie eine Inlinerichtlinie für den Benutzer Dave. Die Richtlinie erteilt dem Benutzer Dave die `s3:PutObject`-Berechtigung. Sie müssen die Richtlinie aktualisieren, indem Sie Ihren Bucket-Namen angeben.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "PermissionForObjectOperations",
         "Effect": "Allow",
         "Action": [
            "s3:PutObject"
         ],
         "Resource": [
            "arn:aws:s3:::awsexamplebucket1/*"
         ]
      }
   ]
}
```

Weitere Informationen finden Sie unter Verwenden von eingebundenen Richtlinien im IAM-Benutzerhandbuch. Beachten Sie, dass Sie sich mit den Anmeldeinformationen von Konto A an der Konsole anmelden müssen.

Schritt 2: Testen der Berechtigungen

Überprüfen Sie unter Verwendung der Anmeldeinformationen von Dave, ob die Berechtigungen funktionieren. Sie haben die Wahl zwischen den folgenden beiden Verfahren.

Testen mit der AWS CLI

1. Aktualisieren Sie die AWS CLI Konfigurationsdatei, indem Sie das folgende UserDaveAccountA-Profil hinzufügen. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

   ```
   [profile UserDaveAccountA]
   aws_access_key_id = access-key
   aws_secret_access_key = secret-access-key
   region = us-east-1
   ```

2. Überprüfen Sie, ob Dave Vorgänge ausführen kann, für die ihm in der Benutzerrichtlinie Berechtigungen erteilt wurden. Laden Sie ein Beispielobjekt mit dem folgenden AWS CLI `put-object` Befehl hoch.

Der Parameter `--body` im Befehl identifiziert die hochzuladende Quelldatei. Befindet sich die Datei beispielsweise im Stammverzeichnis auf dem Laufwerk C: eines Windows-Computers, geben Sie `c:\HappyFace.jpg` an. Der Parameter `--key` gibt den Schlüsselnamen für das Objekt an.

```
aws s3api put-object --bucket awsexamplebucket1 --key HappyFace.jpg --
body HappyFace.jpg --profile UserDaveAccountA
```

Führen Sie den folgenden AWS CLI Befehl aus, um das Objekt abzurufen.

```
aws s3api get-object --bucket awsexamplebucket1 --key HappyFace.jpg OutputFile.jpg
 --profile UserDaveAccountA
```

Testen mit der AWS Tools for Windows PowerShell

1. Speichern Sie die Anmeldeinformationen von Dave als AccountADave. Anschließend verwenden Sie diese Anmeldeinformationen für ein PUT und ein GET für ein Objekt.

   ```
   set-awscredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -storeas
    AccountADave
   ```

2. Laden Sie ein Beispielobjekt mit dem AWS Tools for Windows PowerShell `Write-S3Object` Befehl hoch, indem Sie die gespeicherten Anmeldeinformationen des Benutzers Dave verwenden.

   ```
   Write-S3Object -bucketname awsexamplebucket1 -key HappyFace.jpg -file HappyFace.jpg
    -StoredCredentials AccountADave
   ```

   Laden Sie das zuvor hochgeladene Objekt herunter.

   ```
   Read-S3Object -bucketname awsexamplebucket1 -key HappyFace.jpg -file Output.jpg -
   StoredCredentials AccountADave
   ```

Beispiel 2: Bucket-Eigentümer erteilt kontoübergreifende Bucket-Berechtigungen

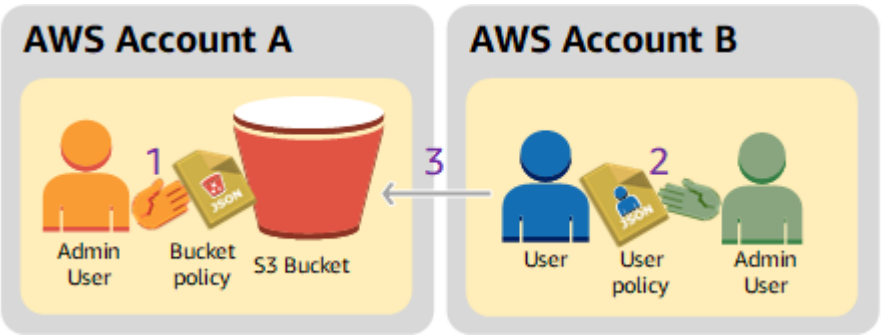
> **Important**
>
> Das Erteilen von Berechtigungen für IAM-Rollen ist eine bessere Vorgehensweise als die Erteilung von Berechtigungen an einzelne Benutzer. Weitere Informationen zur Vorgehensweise finden Sie unter Hintergrund: Kontoübergreifende Berechtigungen und die Verwendung von IAM-Rollen.

Themen

Ein AWS-Konto– z. B. Konto A – kann einem anderen AWS-Konto, Konto B, die Berechtigung erteilen, auf seine Ressourcen wie Buckets und Objekte zuzugreifen. Konto B kann diese Berechtigungen an Benutzer in seinem Konto delegieren. In diesem Beispielszenario erteilt ein Bucket-Eigentümer einem anderen Konto eine kontenübergreifende Berechtigung, um bestimmte Bucket-Vorgänge auszuführen.

> **Note**
>
> Konto A kann einem Benutzer in Konto B unter Verwendung einer Bucket-Richtlinie auch direkt Berechtigungen erteilen. Der Benutzer braucht dennoch eine Berechtigung von seinem übergeordneten Konto, Konto B, zu dem der Benutzer gehört, auch wenn Konto B keine Berechtigungen von Konto A erhalten hat. Solange der Benutzer die Berechtigung von dem Ressourcen-Eigentümer und dem übergeordneten Konto hat, kann der Benutzer auf die Ressource zugreifen.

Nachfolgend finden Sie eine kurze Zusammenfassung der wichtigsten Details:



1. Der Administrator-Benutzer von Konto A ordnet eine Bucket-Richtlinie zu, die Konto B kontenübergreifende Berechtigungen erteilt, um bestimmte Bucket-Vorgänge auszuführen.

   Beachten Sie, dass der Administrator-Benutzer in Konto B die Berechtigungen automatisch erbt.
2. Der Administrator-Benutzer von Konto B ordnet dem Benutzer eine Benutzerrichtlinie zu, die die Berechtigungen an den Benutzer delegiert, die er von Konto A erhalten hat.
3. Der Benutzer in Konto B überprüft die Berechtigungen, indem er auf ein Objekt in dem Bucket zugreift, das Konto A gehört.

Für dieses Beispiel benötigen Sie zwei Konten. Die folgende Tabelle zeigt, wie wir auf diese Konten und die Administrator-Benutzer darin verweisen. Laut den IAM-Richtlinien (siehe Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen) verwenden wir in dieser schrittweisen Anleitung nicht die Anmeldeinformationen des Root-Benutzers. Stattdessen erstellen Sie einen Administrator-Benutzer in jedem Konto, und verwenden dessen Anmeldeinformationen, um Ressourcen zu erstellen und ihnen Berechtigungen zu erteilen.

AWS-Konto ID	Konto bezeichnet als	Administratorbenutzer im Konto
1111-1111-1111	Konto A	AccountAadmin
2222-2222-2222	Konto B	AccountBadmin

Alle Aufgaben in Verbindung mit dem Erstellen von Benutzern und Gewähren von Berechtigungen werden in der AWS Management Console ausgeführt. Um die Berechtigungen zu überprüfen, verwendet die Anleitung die Befehlszeilen-Tools (AWS Command Line Interface CLI) und AWS Tools for Windows PowerShell, sodass Sie keinen Code schreiben müssen.

Schritt 0: Vorbereitung auf den Walkthrough

1. Stellen Sie sicher, dass Sie zwei haben AWS-Konten und dass jedes Konto einen Administratorbenutzer hat, wie in der Tabelle im vorherigen Abschnitt gezeigt.

   a. Melden Sie sich AWS-Konto bei Bedarf für ein an.

   b. Melden Sie sich mit den Anmeldeinformationen für Konto A an der IAM-Konsole an und erstellen Sie wie folgt den Administrator-Benutzer:

      i. Erstellen Sie einen Benutzer AccountAadmin und notieren Sie sich die Sicherheitsanmeldeinformationen. Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.

      ii. Erteilen Sie AccountAadmin Administratorrechte, indem Sie eine Benutzerrichtlinie anfügen, die vollen Zugriff gewährt. Weitere Informationen finden Sie unter Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.

   c. Während Sie in der IAM-Konsole arbeiten, schreiben Sie sich die URL für die Anmeldung des IAM-Benutzers auf dem Dashboard auf. Alle Benutzer in diesem Konto müssen diese URL für die Anmeldung an der AWS Management Console verwenden.

      Weitere Informationen finden Sie unter Wie sich Benutzer in Ihrem Konto anmelden im IAM-Benutzerhandbuch.

   d. Wiederholen Sie den vorherigen Schritt mit den Anmeldeinformationen von Konto B und erstellen Sie den Administratorbenutzer AccountBadmin.

2. Richten Sie entweder die AWS Command Line Interface (CLI) oder die ein AWS Tools for Windows PowerShell. Stellen Sie sicher, dass Sie die Anmeldeinformationen speichern, wie folgt:

   - Wenn Sie die verwenden AWS CLI, erstellen Sie zwei Profile, AccountAadmin und AccountBadmin, in der Konfigurationsdatei.
   - Wenn Sie die verwenden AWS Tools for Windows PowerShell, stellen Sie sicher, dass Sie Anmeldeinformationen für die Sitzung als AccountAadmin und speichern AccountBadmin.

   Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

3. Speichern Sie die Anmeldeinformationen des Administrator-Benutzers, auch als Profile bezeichnet. Sie können den Profilnamen verwenden, statt für jeden eingegebenen Befehl

Anmeldeinformationen anzugeben. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

a. Fügen Sie der AWS CLI Anmeldeinformationsdatei für jeden der Administratorbenutzer in den beiden Konten Profile hinzu.

```
[AccountAadmin]
aws_access_key_id = access-key-ID
aws_secret_access_key = secret-access-key
region = us-east-1

[AccountBadmin]
aws_access_key_id = access-key-ID
aws_secret_access_key = secret-access-key
region = us-east-1
```

b. Wenn Sie die verwenden AWS Tools for Windows PowerShell

```
set-awscredentials -AccessKey AcctA-access-key-ID -SecretKey AcctA-secret-
access-key -storeas AccountAadmin
set-awscredentials -AccessKey AcctB-access-key-ID -SecretKey AcctB-secret-
access-key -storeas AccountBadmin
```

Schritt 1: Erledigen der Aufgaben von Konto A

Schritt 1.1: Anmelden bei der AWS Management Console

Melden Sie sich unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto A zuerst AWS Management Console als AccountAadmin Benutzer bei der an. Dieser Benutzer erstellt einen Bucket und ordnet ihm eine Richtlinie zu.

Schritt 1.2: Erstellen eines Buckets

1. Erstellen Sie in der Amazon-S3-Konsole einen Bucket. Diese Übung geht davon aus, dass der Bucket in der Region USA Ost (Nord-Virginia) erstellt wurde und *DOC-EXAMPLE-BUCKET* heißt.

   Anweisungen finden Sie unter Erstellen eines Buckets.
2. Hochladen eines Beispielobjekts in den Bucket.

   Anweisungen finden Sie unter Schritt 2: Hochladen eines Objekts in Ihren Bucket.

Schritt 1.3: Zuordnen einer Bucket-Richtlinie, um Konto B kontoübergreifende Berechtigungen zu erteilen

Die Bucket-Richtlinie gewährt Konto B die `s3:ListBucket` Berechtigungen `s3:GetLifecycleConfiguration` und . Es wird davon ausgegangen, dass Sie immer noch mit AccountAadmin Benutzeranmeldeinformationen bei der Konsole angemeldet sind.

1. Weisen Sie *DOC-EXAMPLE-BUCKET* die folgende Bucket-Richtlinie zu. Die Richtlinie erteilt Konto B die Berechtigung für die Aktionen `s3:GetLifecycleConfiguration` und `s3:ListBucket`.

   Anweisungen finden Sie unter Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole.

   ```
   {
      "Version": "2012-10-17",
      "Statement": [
         {
            "Sid": "Example permissions",
            "Effect": "Allow",
            "Principal": {
               "AWS": "arn:aws:iam::AccountB-ID:root"
            },
            "Action": [
               "s3:GetLifecycleConfiguration",
               "s3:ListBucket"
            ],
            "Resource": [
               "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            ]
         }
      ]
   }
   ```

2. Überprüfen, ob Konto B (und damit der Administrator-Benutzer) die Vorgänge ausführen kann.

   - Verwenden der AWS CLI

     ```
     aws s3 ls s3://DOC-EXAMPLE-BUCKET --profile AccountBadmin
     aws s3api get-bucket-lifecycle-configuration --bucket DOC-EXAMPLE-BUCKET --
     profile AccountBadmin
     ```

   - Verwenden der AWS Tools for Windows PowerShell

```
get-s3object -BucketName DOC-EXAMPLE-BUCKET -StoredCredentials AccountBadmin
get-s3bucketlifecycleconfiguration -BucketName DOC-EXAMPLE-BUCKET -
StoredCredentials AccountBadmin
```

Schritt 2: Erledigen der Aufgaben von Konto B

Jetzt erstellt der Administrator von Konto B einen Benutzer Dave und delegiert an Dave die Berechtigungen, die er von Konto A erhalten hat.

Schritt 2.1: Anmelden bei der AWS Management Console

Melden Sie sich unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto B zuerst AWS Management Console als AccountBadmin Benutzer bei der an.

Schritt 2.2: Erstellen des Benutzers Dave in Konto B

Erstellen Sie in der IAM-Konsole einen Benutzer, Dave.

Detaillierte Anleitungen finden Sie unter Creating IAM Users (AWS Management Console) (Erstellen von IAM-Benutzern) im IAM-Benutzerhandbuch.

Schritt 2.3: Delegieren von Berechtigungen an den Benutzer Dave

Erstellen Sie mithilfe der folgenden Richtlinie eine Inlinerichtlinie für den Benutzer Dave. Sie müssen die Richtlinie aktualisieren, indem Sie Ihren Bucket-Namen angeben.

Es wird davon ausgegangen, dass Sie mit AccountBadmin Benutzeranmeldeinformationen bei der Konsole angemeldet sind.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example",
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
```

```
        ]
      }
   ]
}
```

Weitere Informationen finden Sie unter Verwenden von eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Schritt 2.4: Testen der Berechtigungen

Jetzt kann Dave in Konto B den Inhalt von *DOC-EXAMPLE-BUCKET* auflisten, der Konto A gehört. Sie können die Berechtigungen mit einem der folgenden Verfahren überprüfen.

Testen mit der AWS CLI

1. Fügen Sie das UserDave Profil der AWS CLI Konfigurationsdatei hinzu. Weitere Informationen zur Config-Datei finden Sie unte Einrichten der Tools für die beispielhaften Walkthroughs.

   ```
   [profile UserDave]
   aws_access_key_id = access-key
   aws_secret_access_key = secret-access-key
   region = us-east-1
   ```

2. Geben Sie an der Eingabeaufforderung den folgenden AWS CLI Befehl ein, um zu überprüfen, ob Dave jetzt eine Objektliste von der abrufen kann, die Konto A *DOC-EXAMPLE-BUCKET* gehört. Beachten Sie, dass der Befehl das UserDave Profil angibt.

   ```
   aws s3 ls s3://DOC-EXAMPLE-BUCKET --profile UserDave
   ```

   Dave besitzt keine anderen Berechtigungen. Wenn er also einen anderen Vorgang versucht, z. B. die folgende Bucket-Lebenszykluskonfiguration abzurufen, gibt Amazon S3 die Berechtigung verweigert zurück.

   ```
   aws s3api get-bucket-lifecycle-configuration --bucket DOC-EXAMPLE-BUCKET --profile
    UserDave
   ```

Testen mit AWS Tools for Windows PowerShell

1. Speichern Sie die Anmeldeinformationen von Dave als AccountBDave.

```
set-awscredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -storeas
 AccountBDave
```

2. Probieren Sie den Befehl List Bucket aus.

```
get-s3object -BucketName DOC-EXAMPLE-BUCKET -StoredCredentials AccountBDave
```

Dave besitzt keine anderen Berechtigungen. Wenn er also einen anderen Vorgang versucht, z. B. die folgende Bucket-Lebenszykluskonfiguration abzurufen, gibt Amazon S3 die Berechtigung verweigert zurück.

```
get-s3bucketlifecycleconfiguration -BucketName DOC-EXAMPLE-BUCKET -
StoredCredentials AccountBDave
```

Schritt 3: (Optional) Versuchen Sie eine explizite Zugriffsverweigerung

Sie können Berechtigungen über eine ACL, eine Bucket-Richtlinie oder eine Benutzerrichtlinie erhalten. Wenn es jedoch eine explizite Zugriffsverweigerung gibt, die entweder über eine Bucket-Richtlinie oder ein Benutzerprofil festgelegt wurde, hat die explizite Zugriffsverweigerung Vorrang gegenüber allen anderen Berechtigungen. Für den Test aktualisieren wir die Bucket-Richtlinie und verweigern Konto B explizit die `s3:ListBucket`-Berechtigung. Die Richtlinie erteilt auch die `s3:ListBucket`-Berechtigung, aber die explizite Zugriffsverweigerung hat Vorragen, und Konto B oder die Benutzer in Konto B können keine Objekte in *DOC-EXAMPLE-BUCKET* auflisten.

1. Ersetzen Sie mithilfe der Anmeldeinformationen des Benutzers AccountAadmin in Konto A die Bucket-Richtlinie durch Folgendes.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example permissions",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:GetLifecycleConfiguration",
```

```
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
         ]
      },
      {
         "Sid": "Deny permission",
         "Effect": "Deny",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
         ]
      }
   ]
}
```

2. Wenn Sie nun versuchen, eine Bucket-Liste mit - AccountBadmin Anmeldeinformationen abzurufen, wird der Zugriff verweigert.

   - Verwenden der AWS CLI:

     ```
     aws s3 ls s3://DOC-EXAMPLE-BUCKET --profile AccountBadmin
     ```

   - Verwenden der AWS Tools for Windows PowerShell:

     ```
     get-s3object -BucketName DOC-EXAMPLE-BUCKET -StoredCredentials AccountBDave
     ```

Schritt 4: Bereinigen

1. Nachdem Sie mit den Tests fertig sind, räumen Sie wie folgt auf.

   - Melden Sie sich mit den Anmeldeinformationen von Konto A bei der AWS Management Console (AWS Management Console) an und gehen Sie wie folgt vor:

- Entfernen Sie in der Amazon-S3-Konsole die an *DOC-EXAMPLE-BUCKET* angefügte Bucket-Richtlinie. Löschen Sie in den Bucket Properties die Richtlinie im Abschnitt Permissions.
- Wenn der Bucket für diese Übung erstellt wurde, löschen Sie in der Amazon-S3-Konsole die Objekte und dann den Bucket.
- Entfernen Sie in der IAM-Konsole den AccountAadmin Benutzer.

2. Melden Sie sich mit den Anmeldeinformationen von Konto B bei der IAM-Konsole an. Löschen Sie den Benutzer AccountBadmin. step-by-step Anweisungen finden Sie unter Löschen eines IAM-Benutzers im IAM-Benutzerhandbuch.

Beispiel 3: Bucket-Eigentümer, der Berechtigungen für Objekte erteilt, die ihm nicht gehören

> **Important**
>
> Das Erteilen von Berechtigungen für IAM-Rollen ist eine bessere Vorgehensweise als die Erteilung von Berechtigungen an einzelne Benutzer. Weitere Informationen zur Vorgehensweise finden Sie unter Hintergrund: Kontoübergreifende Berechtigungen und die Verwendung von IAM-Rollen.

Themen

In dem Szenario für dieses Beispiel will ein Bucket-Eigentümer Berechtigungen für den Zugriff auf Objekte erteilen, aber nicht alle Objekte im Bucket gehören dem Bucket-Eigentümer. In diesem Beispiel versucht der Bucket-Eigentümer, Benutzern in seinem eigenen Konto eine Berechtigung zu erteilen.

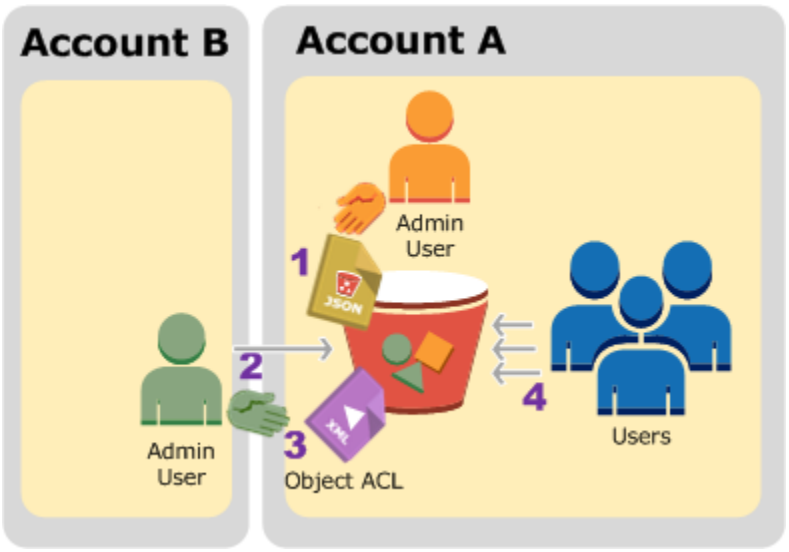
Ein Bucket-Eigentümer kann es anderen ermöglichen AWS-Konten , Objekte hochzuladen. Standardmäßig besitzt der Bucket-Eigentümer keine Objekte, die von einem anderen AWS-

Konto in einen Bucket geschrieben wurden. Objekte gehören den Konten, die sie in einen S3-Bucket schreiben. Wenn der Bucket-Eigentümer keine Objekte im Bucket besitzt, muss der Objekteigentümer dem Bucket-Eigentümer zunächst die Berechtigung mithilfe einer Objekt-ACL erteilen. Anschließend kann der Bucket-Eigentümer einem Objekt, das ihm nicht gehört, Berechtigungen erteilen. Weitere Informationen finden Sie unter Amazon-S3-Bucket- und Objekt-Eigentümerschaft.

Wenn der Bucket-Eigentümer die vom Bucket-Eigentümer erzwungene Einstellung für S3 Object Ownership für den Bucket anwendet, besitzt der Bucket-Eigentümer alle Objekte im Bucket, einschließlich der Objekte, die von einem anderen AWS-Konto geschrieben wurden. Dadurch wird das Problem behoben, dass Objekte nicht im Besitz des Bucket-Eigentümers sind. Anschließend können Sie die Berechtigung an Benutzer in Ihrem eigenen Konto oder an andere AWS-Konten.

> **Note**
>
> S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.
> Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

In diesem Beispiel gehen wir davon aus, dass der Bucket-Eigentümer die vom Bucket-Eigentümer erzwungene Einstellung für Object Ownership nicht angewendet hat. Der Bucket-Eigentümer delegiert die Berechtigung an Benutzer in seinem eigenen Konto. Nachfolgend finden Sie eine kurze Zusammenfassung der wichtigsten Details:



1. Der Administrator-Benutzer von Konto A ordnet eine Bucket-Richtlinie mit zwei Anweisungen zu.
   - Kontoübergreifende Berechtigung für Konto B, um Objekte hochzuladen.
   - Berechtigung für einen Benutzer in seinem eigenen Konto, auf Objekte im Bucket zuzugreifen.
2. Der Administrator-Benutzer für Konto B lädt Objekte in den Bucket hoch, der Konto A gehört.
3. Der Administrator von Konto B aktualisiert die Objekt-ACL, indem er die Berechtigung hinzufügt, die dem Bucket-Eigentümer vollständige Berechtigungen für das Objekt erteilt.
4. Der Benutzer in Konto A überprüft dies durch Zugriff auf Objekte im Bucket, unabhängig davon, wem diese gehören.

Für dieses Beispiel benötigen Sie zwei Konten. Die folgende Tabelle zeigt, wie wir auf diese Konten und die Administrator-Benutzer in diesen Konten verweisen. In dieser schrittweisen Anleitung verwenden Sie die Anmeldeinformationen des Root-Benutzers für das Konto nicht gemäß den empfohlenen IAM-Richtlinien. Weitere Informationen finden Sie unter Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen. Stattdessen erstellen Sie einen Administrator in jedem Konto, und verwenden dessen Anmeldeinformationen, um Ressourcen zu erstellen und ihnen Berechtigungen zu erteilen.

AWS-Konto ID	Konto bezeichnet als	Administrator im Konto
1111-1111-1111	Konto A	AccountAadmin
2222-2222-2222	Konto B	AccountBadmin

Alle Aufgaben in Verbindung mit dem Erstellen von Benutzern und Gewähren von Berechtigungen werden in der AWS Management Console ausgeführt. Um die Berechtigungen zu überprüfen, verwendet die Anleitung die Befehlszeilen-Tools, AWS Command Line Interface (AWS CLI) und AWS Tools for Windows PowerShell, sodass Sie keinen Code schreiben müssen.

Schritt 0: Vorbereitung auf den Walkthrough

1. Stellen Sie sicher, dass Sie zwei haben AWS-Konten und jedes Konto einen Administrator hat, wie in der Tabelle im vorherigen Abschnitt gezeigt.

   a. Melden Sie sich AWS-Konto bei Bedarf für ein an.

   b. Melden Sie sich mit den Anmeldeinformationen von Konto A bei der IAM-Konsole an und führen Sie die folgenden Schritte aus, um einen Administratorbenutzer zu erstellen:

      - Erstellen Sie einen Benutzer AccountAadmin und notieren Sie sich die Sicherheitsanmeldeinformationen. Weitere Informationen zum Hinzufügen von Benutzern finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.
      - Erteilen Sie AccountAadmin Administratorberechtigungen, indem Sie eine Benutzerrichtlinie anfügen, die vollen Zugriff gewährt. Weitere Informationen finden Sie unter Verwalten von IAM-Richtlinien im IAM-Benutzerhandbuch.
      - Notieren Sie sich im Dashboard der IAM-Konsole die URL für die Anmeldung des IAM-Benutzers. Benutzer in diesem Konto müssen diese URL für die Anmeldung an der AWS Management Console verwenden. Weitere Informationen finden Sie unter Wie sich Benutzer in Ihrem Konto anmelden im IAM-Benutzerhandbuch.

   c. Wiederholen Sie den vorherigen Schritt mit den Anmeldeinformationen von Konto B und erstellen Sie den Administratorbenutzer AccountBadmin.

2. Richten Sie entweder die AWS CLI oder die Tools for Windows ein PowerShell. Stellen Sie sicher, dass Sie die Administratoranmeldeinformationen wie folgt speichern:

   - Wenn Sie die verwenden AWS CLI, erstellen Sie zwei Profile, `AccountAadmin` und `AccountBadmin`, in der Konfigurationsdatei.
   - Wenn Sie die Tools for Windows verwenden PowerShell, stellen Sie sicher, dass Sie die Anmeldeinformationen für die Sitzung als `AccountAadmin` und speichern`AccountBadmin`.

   Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

Schritt 1: Erledigen der Aufgaben von Konto A

Führen Sie für Konto A die folgenden Schritte aus:

Schritt 1.1: Anmelden bei der Konsole

Melden Sie sich mit der URL zur Anmeldung von IAM-Benutzern für Konto A bei der AWS Management Console als AccountAadmin Benutzer an. Dieser Benutzer erstellt einen Bucket und ordnet ihm eine Richtlinie zu.

Schritt 1.2: Erstellen eines Buckets und eines Benutzers und Hinzufügen einer Bucket-Richtlinie, um Benutzerberechtigungen zu erteilen

1. Erstellen Sie in der Amazon-S3-Konsole einen Bucket. Diese Übung geht davon aus, dass der Bucket in der Region USA Ost (Nord-Virginia) erstellt wurde und *DOC-EXAMPLE-BUCKET1* heißt.

   Anweisungen finden Sie unter Erstellen eines Buckets.
2. Erstellen Sie in der IAM-Konsole einen Benutzer Dave .

   step-by-step Anweisungen finden Sie unter Erstellen von IAM-Benutzern (Konsole) im IAM-Benutzerhandbuch.
3. Schreiben Sie sich die Anmeldeinformationen für Dave auf.
4. Weisen Sie in der Amazon-S3-Konsole die folgende Bucket-Richtlinie dem *DOC-EXAMPLE-BUCKET1*-Bucket zu. Anweisungen finden Sie unter Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole. Folgen Sie den Schritten, um eine Bucket-Richtlinie hinzuzufügen. Informationen zum Suchen von Konto-IDs finden Sie unter Suchen Ihrer AWS-Konto ID.

   Die Richtlinie erteilt Konto B die Berechtigung `s3:PutObject` und `s3:ListBucket`. Die Richtlinie erteilt außerdem dem Benutzer Dave die `s3:GetObject`-Berechtigung.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Statement1",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
```

```
            "s3:PutObject",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
         ]
      },
      {
         "Sid": "Statement3",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
            "s3:GetObject"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
         ]
      }
   ]
}
```

Schritt 2: Erledigen der Aufgaben von Konto B

Nachdem Konto B die Berechtigung besitzt, Vorgänge für den Bucket von Konto A auszuführen, erledigt der Administrator von Konto B Folgendes:

- Hochladen eines Objekts in den Bucket von Konto A.
- Fügen Sie eine Berechtigung in der Objekt-ACL hinzu, um Account A, dem Bucket-Eigentümer, die volle Kontrolle zu ermöglichen.

Verwenden der AWS CLI

1. Hochladen eines Objekts mit dem CLI-Befehl `put-object`. Der Parameter `-body` im Befehl identifiziert die hochzuladende Quelldatei. Befindet sich die Datei beispielsweise auf dem Laufwerk `C:` eines Windows-Computers, geben Sie `c:\HappyFace.jpg` an. Der Parameter `--key` gibt den Schlüsselnamen für das Objekt an.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET1 --key HappyFace.jpg --body
 HappyFace.jpg --profile AccountBadmin
```

2. Erteilung einer Berechtigung in der Objekt-ACL, um dem Bucket-Eigentümer volle Kontrolle über das Objekt zu erteilen. Informationen darüber, wie Sie eine kanonische Benutzer-ID finden, finden Sie unter Ermitteln der kanonischen Benutzer-ID für Ihr AWS-Konto.

```
aws s3api put-object-acl --bucket DOC-EXAMPLE-BUCKET1 --key HappyFace.jpg --grant-
full-control id="AccountA-CanonicalUserID" --profile AccountBadmin
```

Verwenden der Tools für Windows PowerShell

1. Laden Sie mit den `Write-S3Object` Tools for Windows ein Objekt PowerShellcommandhoch.

```
Write-S3Object -BucketName DOC-EXAMPLE-BUCKET1 -key HappyFace.jpg -file
 HappyFace.jpg -StoredCredentials AccountBadmin
```

2. Erteilung einer Berechtigung in der Objekt-ACL, um dem Bucket-Eigentümer volle Kontrolle über das Objekt zu erteilen.

```
Set-S3ACL -BucketName DOC-EXAMPLE-BUCKET1 -Key HappyFace.jpg -CannedACLName
 "bucket-owner-full-control" -StoredCreden
```

Schritt 3: Testen der Berechtigungen

Überprüfen Sie nun, dass der Benutzer Dave in Konto A Zugriff auf das Objekt hat, das Konto B gehört.

Verwenden der AWS CLI

1. Fügen Sie Benutzer Dave-Anmeldeinformationen zur AWS CLI Konfigurationsdatei hinzu und erstellen Sie ein neues Profil, `UserDaveAccountA`. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

```
[profile UserDaveAccountA]
aws_access_key_id = access-key
aws_secret_access_key = secret-access-key
```

```
region = us-east-1
```

2. Führen Sie den CLI-Befehl `get-object` aus, um `HappyFace.jpg` herunterzuladen und es lokal zu speichern. Sie stellen dem Benutzer Dave Anmeldeinformationen bereit, indem Sie den Parameter `--profile` hinzufügen.

```
aws s3api get-object --bucket DOC-EXAMPLE-BUCKET1 --key
 HappyFace.jpg Outputfile.jpg --profile UserDaveAccountA
```

Verwenden der Tools für Windows PowerShell

1. Speichern Sie die AWS Anmeldeinformationen von Benutzer Dave als im `UserDaveAccountA`persistenten Speicher.

```
Set-AWSCredentials -AccessKey UserDave-AccessKey -SecretKey UserDave-
SecretAccessKey -storeas UserDaveAccountA
```

2. Führen Sie den Befehl `Read-S3Object` aus, um das Objekt `HappyFace.jpg` herunterzuladen und es lokal zu speichern. Sie stellen dem Benutzer Dave Anmeldeinformationen bereit, indem Sie den Parameter `-StoredCredentials` hinzufügen.

```
Read-S3Object -BucketName DOC-EXAMPLE-BUCKET1 -Key HappyFace.jpg -file
 HappyFace.jpg  -StoredCredentials UserDaveAccountA
```

Schritt 4: Bereinigen

1. Nachdem Sie mit den Tests fertig sind, räumen Sie wie folgt auf.

   - Melden Sie sich mit den Anmeldeinformationen von Konto A an der AWS Management Console an und machen Sie Folgendes:

     - Entfernen Sie in der Amazon-S3-Konsole die an *DOC-EXAMPLE-BUCKET1* angefügte Bucket-Richtlinie. Löschen Sie in den Bucket Properties die Richtlinie im Abschnitt Permissions.
     - Wenn der Bucket für diese Übung erstellt wurde, löschen Sie in der Amazon-S3-Konsole die Objekte und dann den Bucket.
     - Entfernen Sie in der IAM-Konsole den AccountAadmin Benutzer. step-by-step Anweisungen finden Sie unter Löschen eines IAM-Benutzers im IAM-Benutzerhandbuch.

2. Melden Sie sich mit den Anmeldeinformationen von Konto B an der AWS Management Console an. Löschen Sie in der IAM-Konsole den Benutzer AccountBadmin.

Beispiel 4: Der Bucket-Eigentümer erteilt eine kontenübergreifende Berechtigung für Objekte, die ihm nicht gehören

Themen

In diesem Beispielszenario besitzen Sie einen Bucket und haben andere aktiviert, AWS-Konten um Objekte hochzuladen. Wenn Sie die vom Bucket-Eigentümer erzwungene Einstellung für S3 Object Ownership für den Bucket angewendet haben, besitzen Sie alle Objekte im Bucket, einschließlich der Objekte, die von einem anderen AWS-Konto geschrieben wurden. Dies behebt das Problem, dass Objekte nicht im Besitz von Ihnen, dem Bucket-Eigentümer, gehören. Anschließend können Sie die Berechtigung an Benutzer in Ihrem eigenen Konto oder an andere AWS-Konten. Angenommen, die erzwungene Einstellung des Bucket-Eigentümers für S3 Object Ownership ist nicht aktiviert. Das heißt, Ihr Bucket kann Objekte enthalten, die anderen AWS-Konten gehören.

Angenommen, Sie müssen als Bucket-Eigentümer einem Benutzer in einem anderen Konto eine kontenübergreifende Berechtigung für Objekte erteilen, unabhängig davon, wer der Eigentümer ist. Dieser Benutzer könnte beispielsweise eine Buchhaltungsanwendung sein, die Zugriff auf Objekt-Metadaten benötigt. Es gibt zwei Kernprobleme:

- Der Bucket-Eigentümer hat keine Berechtigungen für diese Objekte, die von anderen AWS-Konten erstellt wurden. Damit der Bucket-Eigentümer Berechtigungen für Objekte erteilen kann, die ihm nicht gehören, muss der Objekteigentümer, das , das die Objekte erstellt AWS-Konto hat, zuerst dem Bucket-Eigentümer die Berechtigung erteilen. Der Bucket-Eigentümer kann diese Berechtigungen delegieren.

- Das Konto des Bucket-Eigentümers kann Berechtigungen an Benutzer in seinem eigenen Konto delegieren (siehe Beispiel 3: Bucket-Eigentümer, der Berechtigungen für Objekte erteilt, die ihm nicht gehören), aber es kann keine Berechtigungen an andere delegieren AWS-Konten, da die kontoübergreifende Delegierung nicht unterstützt wird.

In diesem Szenario kann der Bucket-Eigentümer eine AWS Identity and Access Management (IAM)-Rolle mit der Berechtigung zum Zugriff auf Objekte erstellen und einem anderen die AWS-Konto Berechtigung erteilen, die Rolle vorübergehend zu übernehmen, sodass er auf Objekte im Bucket zugreifen kann.

> **Note**
>
> S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.
> Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Hintergrund: Kontoübergreifende Berechtigungen und die Verwendung von IAM-Rollen

IAM-Rollen unterstützen verschiedene Szenarien, den Zugriff auf Ihre Ressourcen zu definieren. Der kontenübergreifende Zugriff ist eines der Schlüsselszenarien. In diesem Beispiel verwendet der Bucket-Eigentümer, Konto A, eine IAM-Rolle, um den Objektzugriff vorübergehend kontoübergreifend an Benutzer in einem anderen AWS-Konto, Konto C, zu delegieren. Jeder von Ihnen erstellten IAM-Rolle sind zwei Richtlinien zugeordnet:

- Eine Vertrauensrichtlinie, die ein anderes identifiziert AWS-Konto , das die Rolle übernehmen kann.

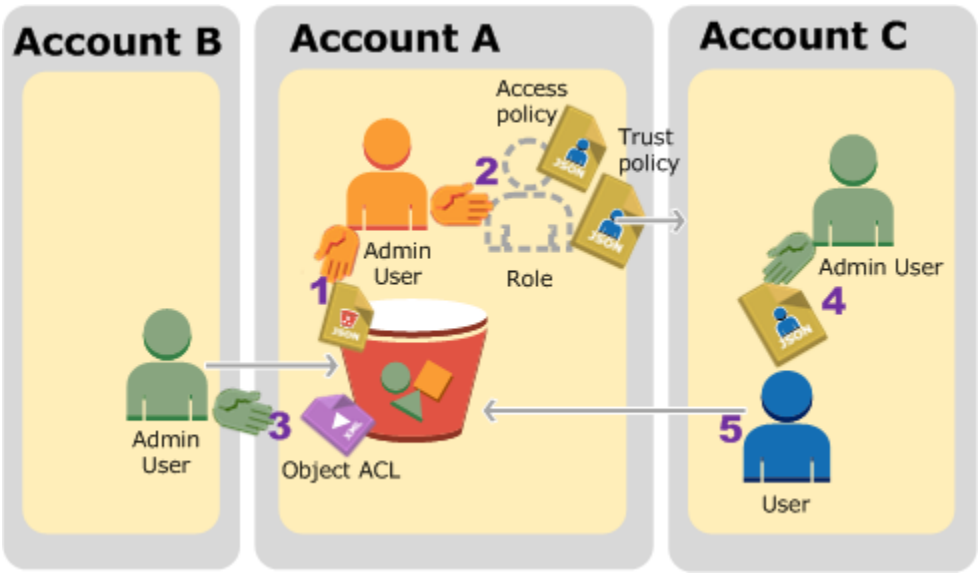
- Eine Zugriffsrichtlinie, die definiert, welche Berechtigungen – z. B. `s3:GetObject` – zulässig sind, wenn jemand die Rolle einnimmt. Eine Liste aller Berechtigungen, die Sie in einer Richtlinie angeben können, finden Sie unter Amazon S3-Richtlinienaktionen.

Das in der Vertrauensrichtlinie AWS-Konto identifizierte erteilt seinem Benutzer dann die Berechtigung, die Rolle zu übernehmen. Der Benutzer kann dann wie folgt auf Objekte zugreifen:

- Die Rolle einnehmen und daraufhin temporäre Sicherheitsanmeldeinformationen erhalten.
- Verwenden der temporären Sicherheitsanmeldeinformationen, um auf die Objekte im Bucket zuzugreifen.

Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

Nachfolgend finden Sie eine kurze Zusammenfassung der wichtigsten Details:



1. Der Administrator-Benutzer von Konto A ordnet eine Bucket-Richtlinie zu, die Konto B die bedingte Berechtigung erteilt, Objekte hochzuladen.
2. Der Administrator von Konto A erstellt eine IAM-Rolle, die eine Vertrauensbeziehung zu Konto C einrichtet, sodass Benutzer in diesem Konto auf Konto A zugreifen können. Die der Rolle zugeordnete Zugriffsrichtlinie beschränkt die Aktionen, die der Benutzer in Konto C machen kann, wenn er auf Konto A zugreift.
3. Der Administrator von Konto B lädt ein Objekt in den Bucket hoch, der Konto A gehört, und erteilt dem Bucket-Eigentümer vollständige Berechtigungen.

4. Der Administrator von Konto C erstellt einen Benutzer und ordnet ihm eine Benutzerrichtlinie zu, die dem Benutzer gestattet, die Rolle zu übernehmen.
5. Der Benutzer in Konto C übernimmt zuerst die Rolle, womit er temporäre Sicherheitsanmeldeinformationen erhält. Unter Verwendung dieser temporären Sicherheitsanmeldeinformationen greift der Benutzer dann auf die Objekte im Bucket zu.

Für dieses Beispiel benötigen Sie drei Konten. Die folgende Tabelle zeigt, wie wir auf diese Konten und die Administrator-Benutzer in diesen Konten verweisen. Laut den IAM-Richtlinien (siehe Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen) verwenden wir in dieser schrittweisen Anleitung nicht die Root-Benutzer des AWS-Kontos -Anmeldeinformationen für das Konto. Stattdessen erstellen Sie einen Administrator-Benutzer in jedem Konto, und verwenden dessen Anmeldeinformationen, um Ressourcen zu erstellen und ihnen Berechtigungen zu erteilen.

AWS-Konto ID	Konto bezeichnet als	Administratorbenutzer im Konto
1111-1111-1111	Konto A	AccountAadmin
2222-2222-2222	Konto B	AccountBadmin
3333-3333-3333	Konto C	AccountCadmin

Schritt 0: Vorbereitung auf den Walkthrough

> **Note**
>
> Öffnen Sie gegebenenfalls einen Texteditor und schreiben Sie sich die Informationen innerhalb der einzelnen Schritte auf. Insbesondere brauchen Sie Konten-IDs, kanonische Benutzer-IDs, URLs für die Anmeldung von IAM-Benutzern für jedes Konto, das mit der Konsole verbunden wird, sowie die Amazon-Ressourcennamen (ARNs) der IAM-Benutzer und -Rollen.

1. Stellen Sie sicher, dass Sie drei haben AWS-Konten und jedes Konto einen Administrator hat, wie in der Tabelle im vorherigen Abschnitt gezeigt.

a. Melden Sie sich bei AWS-Konten Bedarf für an. Wir bezeichnen diese Konten als Konto A, Konto B und Konto C.

b. Melden Sie sich mit den Anmeldeinformationen für Konto A an der IAM-Konsole an und erstellen Sie wie folgt einen Administrator-Benutzer:

   - Erstellen Sie einen Benutzer AccountAadmin und notieren Sie sich die Sicherheitsanmeldeinformationen. Weitere Informationen zum Hinzufügen von Benutzern finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.
   - Erteilen Sie AccountAadmin Administratorrechte, indem Sie eine Benutzerrichtlinie anfügen, die vollen Zugriff gewährt. Weitere Informationen finden Sie unter Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.
   - Schreiben Sie sich auf dem Dashboard der IAM-Konsole die URL für die Anmeldung des IAM-Benutzers auf. Benutzer in diesem Konto müssen diese URL für die Anmeldung an der AWS Management Console verwenden. Weitere Informationen finden Sie unter Wie sich Benutzer in Ihrem Konto anmelden im IAM-Benutzerhandbuch.

c. Wiederholen Sie den obigen Schritt, um Administrator-Benutzer in Konto B und Konto C zu erstellen.

2. Notieren Sie für Konto C die kanonische Benutzer-ID.

   Wenn Sie eine IAM-Rolle in Konto A erstellen, erteilt die Vertrauensrichtlinie Konto C die Berechtigung, die Rolle durch Angabe der Konto-ID zu übernehmen. Sie finden die Konteninformationen wie folgt:

   a. Verwenden Sie Ihre AWS-Konto -ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der Amazon S3-Konsole anzumelden.

   b. Wählen Sie den Namen eines Amazon-S3-Buckets aus, um die Details zu diesem Bucket anzuzeigen.

   c. Klicken Sie auf den Tab Berechtigungen und anschließend auf Access Control List.

   d. Im Abschnitt Access for your AWS-Konto(Zugriff für Ihr AWS-Konto) in der Spalte Konto befindet sich eine lange Kennung, z. B. `c1daexampleaaf850ea79cf0430f33d72579fd1611c97f7ded193374c0b163b6`. Dies ist Ihre kanonische Benutzer-ID.

3. Für das Erstellen einer Bucket-Richtlinie benötigen Sie die folgenden Informationen. Schreiben Sie sich diese Werte auf:

- Kanonische Benutzer-ID von Konto A – Wenn der Administrator von Konto A dem Administrator von Konto B die bedingte Berechtigung erteilt, Objekte hochzuladen, gibt die Bedingung die kanonische Benutzer-ID des Benutzers von Konto A an, der vollständige Kontrolle über die Objekte benötigt.

  > **Note**
  >
  > Die kanonische Benutzer-ID ist ein nur für Amazon S3 geltendes Konzept. Es handelt sich dabei um eine 64 Zeichen lange verschleierte Version der Konto-ID.

- Benutzer-ARN für Administrator von Konto B – Sie finden den Benutzer-ARN in der IAM-Konsole. Sie müssen den Benutzer auswählen und den ARN des Benutzers auf der Registerkarte Zusammenfassung finden.

  In der Bucket-Richtlinie erteilen Sie die AccountBadmin Berechtigung zum Hochladen von Objekten und geben den Benutzer mit dem ARN an. Ein Beispiel für einen ARN-Wert:

  ```
  arn:aws:iam::AccountB-ID:user/AccountBadmin
  ```

4. Richten Sie entweder die AWS Command Line Interface (CLI) oder die ein AWS Tools for Windows PowerShell. Stellen Sie sicher, dass Sie die Anmeldeinformationen speichern, wie folgt:

   - Wenn Sie die verwenden AWS CLI, erstellen Sie die Profile AccountAadmin und AccountBadminin der Konfigurationsdatei.
   - Wenn Sie die verwenden AWS Tools for Windows PowerShell, stellen Sie sicher, dass Sie die Anmeldeinformationen für die Sitzung als AccountAadmin und speichernAccountBadmin.

   Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs.

Schritt 1: Erledigen der Aufgaben von Konto A

In diesem Beispiel ist Konto A der Bucket-Eigentümer. Benutzer AccountAadmin in Konto A erstellt also einen Bucket, fügt eine Bucket-Richtlinie an, die dem Administrator von Konto B die Berechtigung zum Hochladen von Objekten erteilt, erstellt eine IAM-Rolle, die Konto C die Berechtigung erteilt, die Rolle zu übernehmen, damit es auf Objekte im Bucket zugreifen kann.

Schritt 1.1: Anmelden bei der AWS Management Console

Melden Sie sich unter Verwendung der Anmelde-URL des IAM-Benutzers für Konto A zuerst AWS Management Console als AccountAadmin Benutzer bei der an. Dieser Benutzer erstellt einen Bucket und ordnet ihm eine Richtlinie zu.

Schritt 1.2: Erstellen eines Buckets und Anfügen einer Richtlinie

Führen Sie in der Amazon-S3-Konsole die folgenden Schritte aus:

1. Erstellen Sie einen Bucket. Diese Übung setzt voraus, dass der Bucket-Name *DOC-EXAMPLE-BUCKET1* ist.

   Anweisungen finden Sie unter Erstellen eines Buckets.

2. Ordnen Sie die folgende Bucket-Richtlinie zu, die dem Administrator von Konto B die bedingte Berechtigung erteilt, Objekte hochzuladen.

   Sie müssen die Richtlinie aktualisieren, indem Sie Ihre eigenen Werte für *DOC-EXAMPLE-BUCKET1*, *AccountB-ID* und *CanonicalUserId-of-AWSaccountA-BucketOwner* angeben.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "111",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
        },
        {
            "Sid": "112",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
            "Condition": {
```

```
                "StringNotEquals": {
                    "s3:x-amz-grant-full-control": "id=CanonicalUserId-of-
AWSaccountA-BucketOwner"
                }
            }
        }
    ]
}
```

Schritt 1.3: Erstellen einer IAM-Rolle, um Konto C kontenübergreifenden Zugriff in Konto A zu erteilen

Erstellen Sie in der IAM-Konsole eine IAM-Rolle („examplerole“), die Konto C die Berechtigung erteilt, die Rolle zu übernehmen. Stellen Sie sicher, dass sie noch als Administrator von Konto A angemeldet sind, weil die Rolle in Konto A erstellt werden muss.

1. Bevor Sie die Rolle erstellen, richten Sie die verwaltete Richtlinie ein, die die von der Rolle benötigten Berechtigungen definiert. Diese Richtlinie fügen Sie zu einem späteren Zeitpunkt der Rolle an.

   a. Klicken Sie im Navigationsbereich links auf Policies und dann auf Create Policy.

   b. Klicken Sie neben Create Your Own Policy auf Select.

   c. Geben Sie `access-accountA-bucket` in das Feld Policy Name ein.

   d. Kopieren Sie die folgende Zugriffsrichtlinie und fügen Sie sie in das Feld Policy Document ein. Die Zugriffsrichtlinie erteilt der Rolle die Berechtigung `s3:GetObject`, wenn also der Benutzer von Konto C die Rolle übernimmt, kann er nur die Operation `s3:GetObject` ausführen.

      ```
      {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
          }
        ]
      }
      ```

   e. Klicken Sie auf Create Policy.

Die neue Richtlinie wird in der Liste der verwalteten Richtlinien angezeigt.

2. Klicken Sie im Navigationsbereich links auf Roles und dann auf Create New Role.
3. Wählen Sie unter Rollentyp auswählen die Option Rolle für kontoübergreifenden Zugriff aus und klicken Sie dann auf die Schaltfläche Auswählen neben Zugriff zwischen AWS-Konten Ihnen bereitstellen.
4. Geben Sie die Konto-ID von Konto C ein.

   In dieser Schritt-für-Schritt-Anleitung müssen Sie noch nicht die Multi-Factor Authentication (MFA) von den Benutzern verlangen, um die Rolle zu übernehmen. Aktivieren Sie daher diese Option noch nicht.
5. Klicken Sie auf Next Step, um die mit der Rolle verknüpften Berechtigungen einzurichten.
6. Aktivieren Sie das Kontrollkästchen neben der zuvor erstellten Richtlinie `access-accountA-bucket` und klicken Sie auf Next Step.

   Die Prüfseite wird angezeigt, sodass Sie die Einstellungen bestätigen können, bevor Sie die Rolle erstellen. Ein sehr wichtiges, auf dieser Seite zu beachtendes Element ist der Link, den Sie an die Benutzer senden können, die die Rolle verwenden müssen. Die Benutzer, die auf den Link klicken, werden direkt zur Seite Switch Role geleitet, in der die Felder Account ID und Role Name bereits ausgefüllt sind. Dieser Link wird auch auf der Seite Role Summary für beliebige kontoübergreifende Rollen angezeigt.
7. Geben Sie `examplerole` für den Rollennamen ein und klicken Sie dann auf Next Step.
8. Nachdem Sie die Rolle überprüft haben, klicken Sie auf Create Role.

   Die Rolle `examplerole` wird in der Liste der Rollen angezeigt.
9. Klicken Sie auf den Rollennamen `examplerole`.
10. Wählen Sie den Tab Trust Relationships.
11. Klicken Sie auf Show policy document und überprüfen Sie, ob die angezeigte Vertrauensrichtlinie mit der folgenden Richtlinie übereinstimmt.

    Die folgende Vertrauensrichtlinie richtet eine Vertrauensbeziehung zu Konto C ein und gestattet ihm die Aktion `sts:AssumeRole`. Weitere Informationen finden Sie unter AssumeRole in der APIAWS Security Token Service -Referenz zu .

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountC-ID:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

12. Schreiben Sie sich den Amazon-Ressourcennamen (ARN) der von Ihnen erstellten Rolle `examplerole` auf.

    In den nachfolgenden Schritten ordnen Sie eine Benutzerrichtlinie ein, um einem IAM-Benutzer zu erlauben, diese Rolle einzunehmen. Sie identifizieren die Rolle über den ARN-Wert.

Schritt 2: Erledigen der Aufgaben von Konto B

Der Beispiel-Bucket, der Konto A gehört, benötigt Objekte, die anderen Konten gehören. In diesem Schritt lädt der Administrator von Konto B unter Verwendung der Befehlszeilen-Tools ein Objekt hoch.

- Laden Sie mit dem `put-object` AWS CLI Befehl ein Objekt in hoch*DOC-EXAMPLE-BUCKET1*.

  ```
  aws s3api put-object --bucket DOC-EXAMPLE-BUCKET1 --key HappyFace.jpg --
  body HappyFace.jpg --grant-full-control id="canonicalUserId-ofTheBucket0wner" --
  profile AccountBadmin
  ```

  Beachten Sie Folgendes:

  - Der `--Profile` Parameter gibt AccountBadmin das Profil an, sodass das Objekt Konto B gehört.
  - Der Parameter `grant-full-control` erteilt dem Bucket-Eigentümer vollständige Berechtigungen für das Objekt, wie für die Bucket-Richtlinie erforderlich.
  - Der Parameter `--body` identifiziert die hochzuladende Quelldatei. Befindet sich die Datei beispielsweise auf dem Laufwerk C: eines Windows-Computers, geben Sie `c:\HappyFace.jpg` an.

Schritt 3: Erledigen der Aufgaben von Konto C

In den vorigen Schritten hat Konto A bereits eine Rolle erstellt, `examplerole`, die eine Vertrauensbeziehung zu Konto C einrichtet. Aus diesem Grund können die Benutzer in Konto C auf Konto A zugreifen. In diesem Schritt erstellt der Administrator von Konto C einen Benutzer (Dave) und delegiert die Berechtigung `sts:AssumeRole` an ihn, die er von Konto A erhalten hat. Damit kann Dave `examplerole` übernehmen und erhält temporären Zugriff auf Konto A. Die Zugriffsrichtlinie, die Konto A der Rolle zugeordnet hat, beschränkt die Aktionen, die Dave ausführen kann, wenn er auf Konto A zugreift – insbesondere, Objekte in *DOC-EXAMPLE-BUCKET1* hochzuladen.

Schritt 3.1: Erstellen eines Benutzers in Konto C und Delegieren der Berechtigung, **examplerole** anzunehmen

1. Melden Sie sich unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto C zunächst als AccountCadmin Benutzer bei der AWS Management Console an.

2. Erstellen Sie in der IAM-Konsole einen Benutzer, Dave.

   step-by-step Anweisungen finden Sie unter Erstellen von IAM-Benutzern (AWS Management Console) im IAM-Benutzerhandbuch.

3. Schreiben Sie sich die Anmeldeinformationen für Dave auf. Dave benötigt diese Anmeldeinformationen, um die Rolle `examplerole` zu übernehmen.

4. Erstellen Sie eine interne Richtlinie für den IAM-Benutzer Dave, um die Berechtigung `sts:AssumeRole` für Dave für die Rolle `examplerole` in Konto A zu delegieren.

   a. Klicken Sie im linken Navigationsbereich auf Users.

   b. Klicken Sie auf den Benutzernamen Dave.

   c. Wählen Sie auf der Seite mit den Benutzerinformationen den Tab Permissions aus und erweitern Sie den Abschnitt Inline Policies.

   d. Wählen Sie hier klicken (oder Create User Policy).

   e. Klicken Sie auf Custom Policy und dann auf Select.

   f. Geben Sie einen Namen für die Richtlinie in das Feld Policy Name ein.

   g. Kopieren Sie die folgende Richtlinie in das Feld Policy Document:.

      Sie müssen die Richtlinie aktualisieren, indem Sie die ID von Konto A angeben.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sts:AssumeRole"],
      "Resource": "arn:aws:iam::AccountA-ID:role/examplerole"
    }
  ]
}
```

h. Klicken Sie auf Apply Policy.

5. Speichern Sie die Anmeldeinformationen von Dave in der Konfigurationsdatei des , AWS CLI indem Sie ein weiteres Profil hinzufügen, AccountCDave .

```
[profile AccountCDave]
aws_access_key_id = UserDaveAccessKeyID
aws_secret_access_key = UserDaveSecretAccessKey
region = us-west-2
```

Schritt 3.2: Übernehmen der Rolle (examplerole) und Zugreifen auf Objekte

Jetzt kann Dave auf Objekte in dem Bucket zugreifen, der Konto A gehört, nämlich wie folgt:

- Dave übernimmt zuerst die `examplerole` unter Verwendung seiner eigenen Anmeldeinformationen. Damit werden temporäre Anmeldeinformationen zurückgegeben.
- Unter Verwendung der temporären Anmeldeinformationen greift Dave dann auf die Objekte im Bucket von Konto A zu.

1. Führen Sie an der Eingabeaufforderung den folgenden AWS CLI `assume-role` Befehl mit dem Profil AccountCDave aus.

   Sie müssen in dem Befehl den ARN-Wert aktualisieren, indem Sie die ID von Konto A angeben, in dem `examplerole` definiert ist.

```
aws sts assume-role --role-arn arn:aws:iam::accountA-ID:role/examplerole --profile
 AccountCDave --role-session-name test
```

Als Reaktion gibt AWS Security Token Service (STS) temporäre Sicherheitsanmeldeinformationen (Zugriffsschlüssel-ID, geheimer Zugriffsschlüssel und ein Sitzungstoken) zurück.

2. Speichern Sie die temporären Sicherheitsanmeldeinformationen in der AWS CLI Konfigurationsdatei unter dem `TempCred` Profil .

```
[profile TempCred]
aws_access_key_id = temp-access-key-ID
aws_secret_access_key = temp-secret-access-key
aws_session_token = session-token
region = us-west-2
```

3. Führen Sie an der Eingabeaufforderung den folgenden AWS CLI Befehl aus, um mit den temporären Anmeldeinformationen auf Objekte zuzugreifen. Beispielsweise gibt der Befehl die Head-Objekt API an, um die Objekt-Metadaten für das Objekt `HappyFace.jpg` abzurufen.

```
aws s3api get-object --bucket DOC-EXAMPLE-BUCKET1 --
key HappyFace.jpg SaveFileAs.jpg --profile TempCred
```

Die `examplerole` zugeordnete Zugriffsrichtlinie erlaubt die Aktionen, deshalb verarbeitet Amazon S3 die Anforderung. Sie können das mit jeder anderen Aktion für jedes andere Objekt im Bucket ausprobieren.

Wenn Sie eine andere Aktion ausprobieren, z. B. `get-object-acl`, wird die Berechtigung verweigert, weil die Rolle diese Aktion nicht ausführen darf.

```
aws s3api get-object-acl --bucket DOC-EXAMPLE-BUCKET1 --key HappyFace.jpg --profile
 TempCred
```

Wir haben den Benutzer Dave verwendet, um die Rolle zu übernehmen und unter Verwendung temporärer Anmeldeinformationen auf das Objekt zuzugreifen. Es könnte auch eine Anwendung in Konto C sein, die auf Objekte in *DOC-EXAMPLE-BUCKET1* zugreift. Die Anwendung kann temporäre Anmeldeinformationen erhalten, und Konto C kann die Berechtigung der Anwendung delegieren, um `examplerole` zu übernehmen.

Schritt 4: Bereinigen

1. Nachdem Sie mit den Tests fertig sind, räumen Sie wie folgt auf.

    - Melden Sie sich mit den Anmeldeinformationen von Konto A bei der AWS Management Console (AWS Management Console) an und gehen Sie wie folgt vor:

        - Entfernen Sie in der Amazon-S3-Konsole die an *DOC-EXAMPLE-BUCKET1* angefügte Bucket-Richtlinie. Löschen Sie in den Bucket Properties die Richtlinie im Abschnitt Permissions.
        - Wenn der Bucket für diese Übung erstellt wurde, löschen Sie in der Amazon-S3-Konsole die Objekte und dann den Bucket.
        - Entfernen Sie im https://console.aws.amazon.com/iam/ die , die `examplerole` Sie in Konto A erstellt haben. step-by-step Anweisungen finden Sie unter Löschen eines IAM-Benutzers im IAM-Benutzerhandbuch.
        - Entfernen Sie in der https://console.aws.amazon.com/iam/ den AccountAadmin Benutzer.
2. Melden Sie sich mit den Anmeldeinformationen von Konto B bei der IAM-Konsole an. Löschen Sie den Benutzer AccountBadmin.
3. Melden Sie sich mit den Anmeldeinformationen von Konto C bei der IAM-Konsole an. Löschen Sie AccountCadmin und den Benutzer Dave.

Zugehörige Ressourcen

- Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer im IAM-Benutzerhandbuch.
- Tutorial: Delegieren des Zugriffs in allen AWS-Konten mithilfe von IAM-Rollen im IAM-Benutzerhandbuch.
- Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.

Verwenden von serviceverknüpften Rollen für Amazon S3 Storage Lens

Um Amazon S3 Storage Lens zum Sammeln und Aggregieren von Metriken für alle Ihre Konten in AWS Organizations-Organisationen zu verwenden, müssen Sie zunächst sicherstellen, dass S3-Storage-Lens über vertrauenswürdigen Zugriff verfügt, der durch das Verwaltungskonto in Ihrer Organisation aktiviert ist. S3 Storage Lens erstellt eine serviceverknüpfte Rolle, damit es die Liste der zu Ihrer Organisation AWS-Konten gehörenden abrufen kann. Diese Liste von Konten wird von S3

Storage Lens verwendet, um Metriken für S3-Ressourcen in allen Mitgliedskonten zu sammeln, wenn das S3-Storage-Lens-Dashboard oder die Konfigurationen erstellt oder aktualisiert werden.

Amazon S3 Storage Lens verwendet AWS Identity and Access Management (IAM) serviceverknüpfte Rollen . Eine serviceverknüpfte Rolle ist ein spezieller Typ von IAM-Rolle, die direkt mit S3 Storage Lens verknüpft ist. Serviceverknüpfte Rollen werden von S3 Storage Lens vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer - AWS Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle vereinfacht das Einrichten von S3 Storage Lens, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. S3 Storage Lens definiert die Berechtigungen seiner serviceverknüpften Rollen. Sofern keine andere Konfiguration festgelegt wurde, kann nur S3 Storage Lens die Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können diese serviceverknüpfte Rolle erst löschen, nachdem die zugehörigen Ressourcen gelöscht wurden. Dies schützt Ihre S3-Storage-Lens-Ressourcen, da Sie nicht versehentlich die Berechtigung für den Zugriff auf die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Yes (Ja) in der Spalte Service-Linked Role (Serviceverknüpfte Rolle) angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Berechtigungen für serviceverknüpfte Rollen für Amazon S3 Storage Lens

S3 Storage Lens verwendet die serviceverknüpfte Rolle namens AWSServiceRoleForS3StorageLens – Dies ermöglicht den Zugriff auf AWS Services und Ressourcen, die von S3 Storage Lens verwendet oder verwaltet werden. Auf diese Weise kann S3 Storage Lens in Ihrem Namen auf - AWS Organizations Ressourcen zugreifen.

Die serviceverknüpfte Rolle S3 Storage Lens vertraut dem folgenden Service auf dem Speicher Ihres Unternehmens:

- `storage-lens.s3.amazonaws.com`

Die Rollenberechtigungsrichtlinie erlaubt S3 Storage Lens die Durchführung der folgenden Aktionen:

- `organizations:DescribeOrganization`

```
organizations:ListAccounts

organizations:ListAWSServiceAccessForOrganization

organizations:ListDelegatedAdministrators
```

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für S3 Storage Lens

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie eine der folgenden Aufgaben ausführen, während Sie bei der - AWS Organizations Verwaltung oder den delegierten Administratorkonten angemeldet sind, erstellt S3 Storage Lens die serviceverknüpfte Rolle für Sie:

- Erstellen Sie in der Amazon-S3-Konsole eine S3-Storage-Lens-Dashboard-Konfiguration für Ihr Unternehmen.
- PUT eine S3-Storage-Lens-Konfiguration für Ihre Organisation unter Verwendung der REST API AWS CLI und der SDKs .

> **Note**
>
> S3 Storage Lens wird maximal fünf delegierte Administratoren pro Unternehmen unterstützen.

Wenn Sie diese serviceverknüpfte Rolle löschen, werden sie von den vorherigen Aktionen bei Bedarf neu erstellt.

Beispielrichtlinie für die serviceverknüpfte Rolle S3 Storage Lens

Example Berechtigungsrichtlinie für die serviceverknüpfte Rolle S3 Storage Lens

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AwsOrgsAccess",
            "Effect": "Allow",
            "Action": [
                "organizations:DescribeOrganization",
                "organizations:ListAccounts",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:ListDelegatedAdministrators"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Bearbeiten einer serviceverknüpften Rolle für Amazon S3 Storage Lens

S3 Storage Lens erlaubt es Ihnen nicht, die AWSServiceRoleForS3StorageLens serviceverknüpfte Rolle zu bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon S3 Storage Lens

Wenn Sie die serviceverknüpfte Rolle nicht mehr verwenden müssen, empfehlen wir, die Rolle zu löschen. Auf diese Weise haben Sie keine ungenutzte Entität, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch die Ressourcen für Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

> **Note**
>
> Wenn der Amazon-S3-Storage-Lens-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt das Löschen möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

Um die zu löschen, müssen AWSServiceRoleForS3StorageLens Sie alle S3 Storage Lens-Konfigurationen auf Organisationsebene löschen, die in allen Regionen vorhanden sind, indem Sie die - AWS Organizations Verwaltung oder die delegierten Administratorkonten verwenden.

Die Ressourcen sind S3-Storage-Lens-Konfigurationen auf Organisationsebene. Verwenden Sie S3 Storage Lens, um die Ressourcen zu bereinigen, und verwenden Sie dann die IAM-Konsole , CLI, REST API oder AWS SDK, um die Rolle zu löschen.

In der REST-API AWS CLI und den SDKs können S3-Storage-Lens-Konfigurationen mithilfe von `ListStorageLensConfigurations` in allen Regionen erkannt werden, in denen Ihre Organisation S3-Storage-Lens-Konfigurationen erstellt hat. Verwenden Sie die Aktion `DeleteStorageLensConfiguration`, um diese Konfigurationen zu löschen, damit Sie die Rolle dann löschen können.

> **Note**
>
> Um die serviceverknüpfte Rolle zu löschen, müssen Sie alle S3-Storage-Lens-Konfigurationen auf Organisationsebene in allen Regionen löschen, in denen sie existieren.

So löschen Sie Amazon S3-Storage-Lens-Ressourcen, die von der verwendet werden AWSServiceRoleForS3StorageLens

1. Sie müssen die `ListStorageLensConfigurations` in jeder Region verwenden, in der Sie S3-Storage-Lens-Konfigurationen haben, um eine Liste Ihrer Konfigurationen auf Organisationsebene zu erhalten. Diese Liste kann auch von der Amazon-S3-Konsole bezogen werden.
2. Diese Konfigurationen müssen dann von den entsprechenden regionalen Endpunkten gelöscht werden, indem der API-Aufruf `DeleteStorageLensConfiguration` aufgerufen wird oder über die Amazon-S3-Konsole.

So löschen Sie die serviceverknüpfte Rolle mit IAM

Nachdem Sie die Konfigurationen gelöscht haben, löschen Sie die AWSServiceRoleForS3StorageLens aus der IAM-Konsole oder durch Aufrufen der IAM-API `DeleteServiceLinkedRole`oder mithilfe der AWS CLI oder des AWS SDK. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte Rollen mit S3 Storage Lens

S3 Storage Lens unterstützt die Verwendung von serviceverknüpften Rollen in allen , in AWS-Regionen denen der Service verfügbar ist. Weitere Informationen finden Sie unter Amazon-S3-Regionen und Endpunkte.

AWS Von verwaltete Richtlinien für Amazon S3

Eine AWS von verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet wird AWS. Von AWS verwaltete Richtlinien sind so konzipiert, dass sie Berechtigungen für viele häufige Anwendungsfälle bereitstellen, sodass Sie mit der Zuweisung von Berechtigungen für Benutzer, Gruppen und Rollen beginnen können.

Beachten Sie, dass von AWS verwaltete Richtlinien möglicherweise keine Berechtigungen mit den geringsten Berechtigungen für Ihre spezifischen Anwendungsfälle gewähren, da sie für alle - AWS Kunden verfügbar sind. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie kundenverwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in verwalteten AWS Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS von verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert, wirkt sich die Aktualisierung auf alle Prinzipalidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie angefügt ist. aktualisiert am AWS wahrscheinlichsten eine von AWS verwaltete Richtlinie, wenn ein neuer gestartet AWS-Service wird oder neue API-Operationen für vorhandene Services verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS Von verwaltete Richtlinie: AmazonS3FullAccess

Sie können die `AmazonS3FullAccess`-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt Berechtigungen, die vollen Zugriff auf Amazon S3 ermöglichen.

Informationen zum Anzeigen der Berechtigungen für diese Richtlinie finden Sie unter AmazonS3FullAccess im AWS Management Console.

AWS Von verwaltete Richtlinie: AmazonS3ReadOnlyAccess

Sie können die `AmazonS3ReadOnlyAccess`-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt Berechtigungen, die einen schreibgeschützten Zugriff auf Amazon S3 erlauben.

Informationen zum Anzeigen der Berechtigungen für diese Richtlinie finden Sie unter AmazonS3ReadOnlyAccess im AWS Management Console.

AWS Von verwaltete Richtlinie: AmazonS3ObjectLambdaExecutionRolePolicy

Stellt AWS Lambda Funktionen die erforderlichen Berechtigungen zum Senden von Daten an S3 Object Lambda bereit, wenn Anforderungen an einen S3 Object Lambda-Zugriffspunkt gestellt werden. Gewährt Lambda auch Berechtigungen zum Schreiben in Amazon- CloudWatch Protokolle.

Informationen zum Anzeigen der Berechtigungen für diese Richtlinie finden Sie unter AmazonS3ObjectLambdaExecutionRolePolicy im AWS Management Console.

Amazon S3-Updates für - AWS verwaltete Richtlinien

Anzeigen von Details zu Aktualisierungen für - AWS verwaltete Richtlinien für Amazon S3, seit dieser Service mit der Verfolgung dieser Änderungen begonnen hat.

Änderung	Beschreibung	Datum
Amazon S3 hat die Beschreibungsberechtigungen zu `AmazonS3ReadOnlyAccess` hinzugefügt.	Amazon S3 hat die `s3:Describe*` -Berechtigungen zu `AmazonS3ReadOnlyAccess` hinzugefügt.	11. August 2023
Amazon S3 hat S3 Objekt Lambda-Berechtigungen zu `AmazonS3FullAccess` und `AmazonS3ReadOnlyAccess` hinzugefügt	Amazon S3 hat die `AmazonS3FullAccess` - und `AmazonS3ReadOnlyAccess` -Richtlinien zur Einbeziehung von Berechtigungen für S3-Objekt-Lambda aktualisiert.	27. September 2021
Amazon S3 `AmazonS3ObjectLambdaExecutionRolePolicy` hinzugefügt	Amazon S3 hat eine neue AWS-verwaltete Richtlinie namens hinzugefügt`AmazonS3ObjectLambdaExecutionRolePolicy` , die Lambda-Funktionen Berechtigungen für die Interaktion mit S3 Object	18. August 2021

Änderung	Beschreibung	Datum
	Lambda und das Schreiben in CloudWatch Protokolle bietet.	
Amazon S3 hat mit der Verfolgung von Änderungen begonnen	Amazon S3 hat mit der Verfolgung von Änderunge n für seine - AWS verwaltete Richtlinien begonnen.	18. August 2021

Verwalten des Zugriffs mit S3-Zugriffsberechtigungen

Zur Einhaltung des Grundsatzes der geringsten Berechtigung definieren Sie die Details des Zugriffs auf Ihre Amazon-S3-Daten basierend auf Anwendungen, Personas, Gruppen oder Organisationseinheiten. Abhängig von Umfang und Komplexität der Zugriffsmuster können Sie verschiedene Ansätze zur Gewährung eines detaillierten Zugriffs auf Ihre Daten in Amazon S3 verwenden.

Der einfachste Ansatz für die Verwaltung des Zugriffs auf eine small-to-medium Anzahl von Datensätzen in Amazon S3 durch AWS Identity and Access Management (IAM)-Prinzipale besteht darin, IAM-Berechtigungsrichtlinien und S3-Bucket-Richtlinien zu definieren. Diese Strategie funktioniert, solange die notwendigen Richtlinien innerhalb der Größenbeschränkungen der S3-Bucket-Richtlinien (20 KB) und der IAM-Richtlinien (5 KB) und innerhalb der Anzahl der pro Konto zulässigen IAM-Prinzipale liegen.

Wenn die Zahl Ihrer Datensätze und Anwendungsfälle wächst, werden auch Richtlinien wichtiger. Ein Ansatz, der Richtlinienanweisungen deutlich mehr Raum gibt, besteht in der Verwendung von S3-Zugangspunkten als zusätzlichen Endpunkten für S3-Buckets, da jeder Zugangspunkt eine eigene Richtlinie haben kann. Sie können ziemlich detaillierte Zugriffskontrollmuster definieren, da Sie Tausende von Zugriffspunkten pro AWS-Region und Konto haben können, mit einer Richtlinie von bis zu 20 KB für jeden Zugriffspunkt. S3-Zugangspunkte vergrößern zwar den verfügbaren Raum für Richtlinien, erfordern jedoch einen Mechanismus, mit dem Clients den richtigen Zugangspunkt für den richtigen Datensatz finden können.

Ein dritter Ansatz besteht in der Implementierung eines Musters mit IAM-Session-Brokern. Hier implementieren Sie eine Logik für die Zugriffsentscheidung und generieren für jede Zugriffssitzung dynamisch kurzfristige Anmeldeinformationen für IAM-Sitzungen. Der Ansatz mit IAM-

Sitzungsbrokern unterstützt willkürlich dynamische Berechtigungsmuster und kann effektiv skaliert werden. Sie müssen jedoch die Zugriffsmusterlogik erstellen.

Sie können anstelle dieser Ansätze S3 Access Grants verwenden, um den Zugriff auf Ihre Amazon-S3-Daten zu verwalten. S3 Access Grants bietet ein vereinfachtes Modell für die Definition von Zugriffsberechtigungen für Daten in Amazon S3 nach Präfix, Bucket oder Objekt. Darüber hinaus können Sie S3 Access Grants verwenden, um IAM-Prinzipalen und Benutzern oder Gruppen im Unternehmensverzeichnis Zugriff zu gewähren.

In der Regel definieren Sie Berechtigungen für Daten in Amazon S3, indem Sie Benutzer und Gruppen Datensätzen zuordnen. Sie können mit S3 Access Grants die direkte Zuordnung von S3-Präfixen zu Benutzern und Rollen in Amazon-S3-Buckets und -Objekten definieren. Mit dem vereinfachten Zugriffsschema in S3 Access Grants können Sie IAM-Prinzipalen und Benutzern oder Gruppen im Unternehmensverzeichnis Lese-, Schreib- oder Lese-Schreib-Zugriff auf S3-Präfix-Basis gewähren. Über diese S3-Access-Grants-Funktionen können Anwendungen im Namen des aktuell authentifizierten Benutzers der Anwendung Daten von Amazon S3 anfordern.

Wenn Sie S3 Access Grants mit der Funktion zur Weitergabe vertrauenswürdiger Identitäten von integrieren AWS IAM Identity Center, können Ihre Anwendungen Anforderungen an AWS-Services (einschließlich S3 Access Grants) direkt im Namen eines authentifizierten Unternehmensverzeichnisbenutzers stellen. Ihre Anwendungen müssen den Benutzer nicht mehr zuerst einem IAM-Prinzipal zuordnen. Darüber hinaus wird die Überprüfung vereinfacht, welcher Benutzer auf welches S3-Objekt zugegriffen hat, da Endbenutzeridentitäten bis zu Amazon S3 weitergegeben werden. Sie müssen die Beziehung zwischen Benutzern und IAM-Sitzungen nicht mehr rekonstruieren. Wenn Sie S3 Access Grants mit der Weitergabe vertrauenswürdiger Identitäten im IAM Identity Center kombinieren, enthält jedes AWS CloudTrail-Datenereignis für Amazon S3 einen direkten Verweis auf den Endbenutzer, in dessen Namen auf die Daten zugegriffen wurde.

Weitere Informationen zu S3 Access Grants finden Sie in den folgenden Themen.

Themen

- Konzepte von S3 Access Grants
- S3 Access Grants und Unternehmensverzeichnisidentitäten
- Erste Schritte mit S3 Access Grants
- Erstellen einer S3-Access-Grants-Instance
- Registrieren eines Speicherorts
- Erstellen von Gewährungen

- Anfordern des Zugriffs auf Amazon-S3-Daten über S3 Access Grants
- Zugriff auf S3-Daten über eine Zugriffsgewährung
- Kontoübergreifender Zugriff mit S3 Access Grants
- Verwenden von AWS Tags mit S3 Access Grants
- Limits für S3 Access Grants
- S3-Access-Grants-Integrationen

Konzepte von S3 Access Grants

S3 Access Grants führt die folgenden Konzepte für das vereinfachte Zugriffsschema ein:

S3-Access-Grants-Instances

Eine S3 Access Grants-Instance ist ein logischer Container für einzelne Gewährungen, die definieren, wer welchen Zugriff auf welche Amazon-S3-Daten hat. Es kann pro AWS-Region und AWS-Konto eine S3-Access-Grants-Instance geben. Sie verwenden diese S3-Access-Grants-Instance, um den Zugriff auf alle Buckets im selben Konto und in derselben zu steuern AWS-Region. Wenn Sie S3 Access Grants verwenden möchten, um Benutzer- und Gruppenidentitäten in Ihrem Unternehmensverzeichnis Zugriff zu gewähren, müssen Sie Ihre S3-Access-Grants-Instance auch einer AWS Identity and Access Management (IAM)-Identity-Center-Instance zuordnen.

Orte

Ein Ort definiert, auf welche Daten Ihre S3-Access-Grants-Instance Zugriff gewähren kann. S3 Access Grants gibt IAM-Anmeldeinformationen mit Zugriff auf bestimmte S3-Präfixe, -Buckets oder -Objekte aus. Sie ordnen einen S3-Access-Grants-Ort einer IAM-Rolle zu, über die diese temporären Sitzungen erstellt werden. Die häufigste Konfiguration für einen Ort ist ein einzelner Ort in `s3://` für die gesamte S-Access-Grants-Instance, die den Zugriff auf alle S3-Buckets im Konto und in der AWS-Region abdecken kann. Sie können in Ihrer S3-Access-Grants-Instance auch mehrere Orte erstellen. Sie können beispielsweise einen Bucket als Ort `s3://DOC-EXAMPLE-BUCKET1` für Gewährungen registrieren, die Sie auf diesen Bucket einschränken möchten. Sie können auch den Standardort `s3://` registrieren.

Gewährungen

Um den Zugriffsumfang innerhalb eines Orts einzuschränken, erstellen Sie individuelle Gewährungen. Eine individuelle Gewährung in einer S3-Access-Grants-Instance ermöglicht

bestimmten Entitäten, z. B. IAM-Prinzipalen oder Benutzern bzw. Gruppen in einem Unternehmensverzeichnis, Zugriff auf ein Amazon-S3-Präfix, einen Bucket oder ein Objekt. Sie können für jede Gewährung einen anderen Bereich (ein Präfix, einen Bucket oder ein Objekt) und eine andere Zugriffsebene (`READ`, `WRITE` oder `READWRITE`) definieren. Beispielsweise könnte es eine Gewährung geben, die einer bestimmten Unternehmensverzeichnisgruppe `01234567-89ab-cdef-0123-456789abcdef` den `READ`-Zugriff auf `s3://DOC-EXAMPLE-BUCKET1/projects/items/*` gewährt. Diese Gewährung gibt Benutzern in dieser Gruppe `READ`-Zugriff auf alle Objekte mit einem Schlüsselnamen mit dem Präfix `projects/items/` im Bucket mit dem Namen *DOC-EXAMPLE-BUCKET1*.

Temporäre S3-Access-Grants-Anmeldeinformationen

Eine Anwendung kann just-in-time Zugriffsanmeldeinformationen anfordern, indem sie eine neue S3-API-Operation aufruftGetDataAccess, um den Zugriff auf ein einzelnes Objekt, Präfix oder einen Bucket mit der Berechtigungsstufe `READ`, `WRITE`oder anzufordern`READWRITE`. Die S3-Access-Grants-Instance evaluiert die `GetDataAccess`-Anforderung anhand der verfügbaren Gewährungen. Wenn es eine übereinstimmende Gewährung gibt, übernimmt S3 Access Grants die IAM-Rolle, die dem Ort der übereinstimmenden Gewährung zugeordnet ist. S3 Access Grants richtet anschließend die Berechtigungen der IAM-Sitzung exakt auf die S3-Buckets, Präfixe oder Objekte aus, die im Umfang der Gewährung angegeben sind. Die Ablaufzeit der temporären Anmeldeinformationen für den Zugriff ist standardmäßig auf 1 Stunde festgelegt. Sie können sie jedoch auf einen beliebigen Wert zwischen 15 Minuten und 36 Stunden festlegen.

Funktionsweise

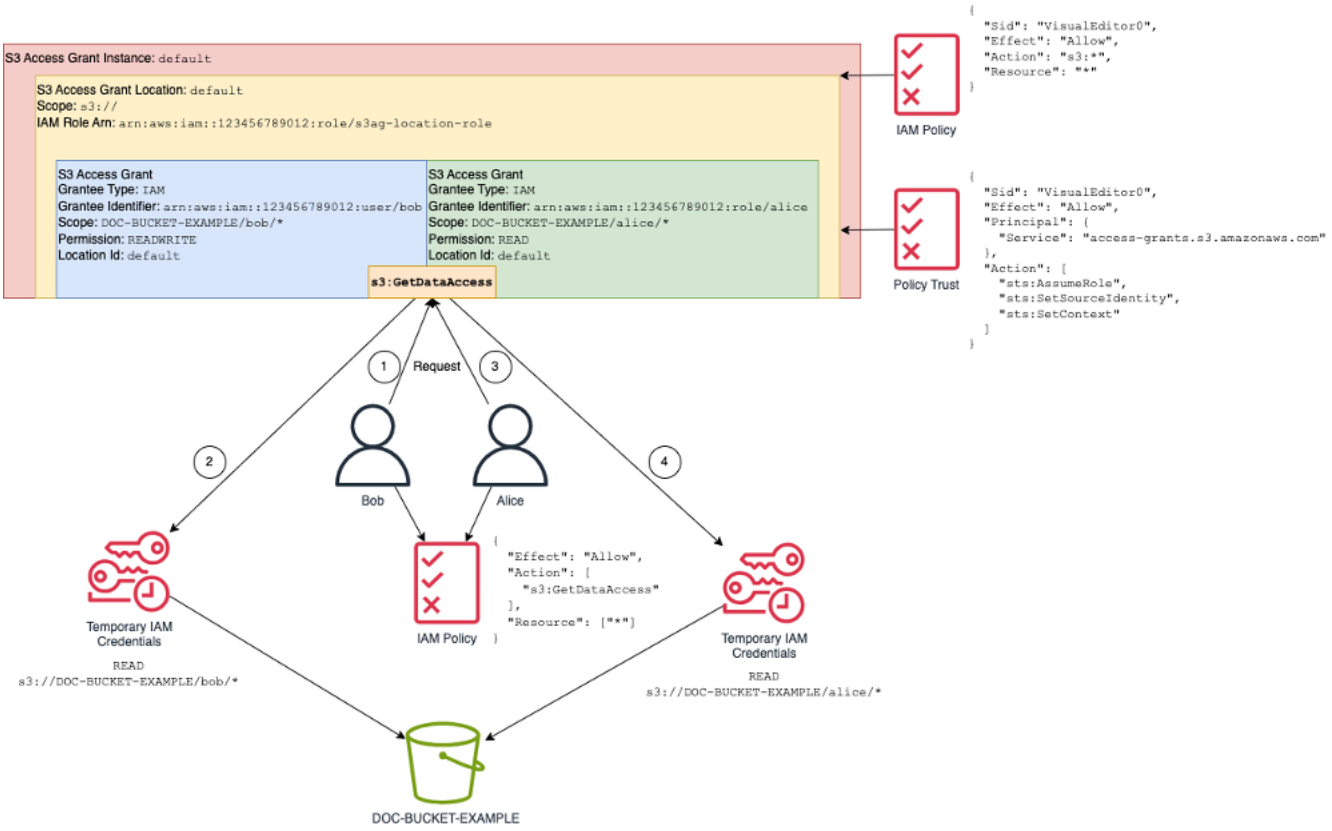
Im folgenden Diagramm ist ein Amazon-S3-Standardort mit dem Umfang `s3://` für die IAM-Rolle `s3ag-location-role` registriert. Diese IAM-Rolle ist berechtigt, im Konto Amazon-S3-Aktionen auszuführen, wenn ihre Anmeldeinformationen über S3 Access Grants abgerufen werden.

An diesem Ort werden zwei verschiedene Zugriffsgewährungen für zwei IAM-Benutzer erstellt. Dem IAM-Benutzer Bob wird sowohl `READ`-Zugriff als auch `WRITE`-Zugriff auf das Präfix `bob/` im Bucket `DOC-BUCKET-EXAMPLE` gewährt. Eine weitere IAM-Rolle, Alice, erhält nur `READ` Zugriff auf das `alice/` Präfix im `DOC-BUCKET-EXAMPLE` Bucket. Es wurde eine Gewährung (blau) definiert, damit Bob auf das Präfix `bob/` im Bucket `DOC-BUCKET-EXAMPLE` zugreifen kann. Es wurde eine Gewährung (grün) definiert, damit Alice auf das Präfix `alice/` im Bucket `DOC-BUCKET-EXAMPLE` zugreifen kann.

Wenn Bob `READ` Daten abrufen kann, ruft die IAM-Rolle, die dem Speicherort zugeordnet ist, an dem sich seine Erteilung befindet, die S3-Access-GrantsGetDataAccess-API-Operation

auf. Wenn Bob versucht, READ-Zugriff für S3-Präfixe oder Objekte zu erhalten, die mit `s3://DOC-BUCKET-EXAMPLE/bob/*` beginnen, gibt die `GetDataAccess`-Anforderung temporärer Anmeldeinformationen für eine IAM-Sitzung mit der entsprechenden Berechtigung für `s3://DOC-BUCKET-EXAMPLE/bob/*` zurück. Ähnlich kann Bob `WRITE`-Zugriff für alle S3-Präfixe oder Objekte erhalten, die mit `s3://DOC-BUCKET-EXAMPLE/bob/*` beginnen, da die Gewährung dies ebenfalls zulässt.

Und ähnlich kann Alice READ-Zugriff für alles erhalten, das mit `s3://DOC-BUCKET-EXAMPLE/alice/` beginnt. Wenn sie jedoch versucht, `WRITE`-Zugriff auf Buckets, Präfixe oder Objekte in `s3://` zu erhalten, wird ihr die Fehlermeldung „Access Denied“ (403 Forbidden) angezeigt, da es keine Gewährung gibt, die ihr einen `WRITE`-Zugriff auf Daten gewährt. Wenn Alice Zugriff (`READ` oder `WRITE`) auf Daten außerhalb von `s3://DOC-BUCKET-EXAMPLE/alice/` anfordert, wird ihr ebenfalls die Fehlermeldung „Access Denied“ angezeigt.



Dieses Muster kann auf eine große Zahl von Benutzern und Buckets skaliert werden und vereinfacht die Verwaltung dieser Berechtigungen. Anstatt jedes Mal, wenn Sie individuelle Benutzer-Präfix-Zugriffsbeziehungen hinzufügen oder entfernen möchten, potenziell umfangreiche S3-Bucket-Richtlinien zu bearbeiten, können Sie einzelne, diskrete Gewährungen hinzufügen und entfernen.

S3 Access Grants und Unternehmensverzeichnisidentitäten

Sie können Amazon S3 Access Grants verwenden, um AWS Identity and Access Management (IAM)-Prinzipalen (Benutzern oder Rollen) Zugriff zu gewähren, sowohl im selben AWS-Konto als auch in anderen. Häufig handelt es sich jedoch bei der Entität, die auf die Daten zugreift, um Endbenutzer in Ihrem Unternehmensverzeichnis. Anstatt IAM-Prinzipalen Zugriff zu gewähren, können Sie mit S3 Access Grants Unternehmensbenutzern und -gruppen direkt Zugriff gewähren. Mit S3 Access Grants müssen Sie Ihre Unternehmensidentitäten nicht mehr zwischengeschalteten IAM-Prinzipalen zuordnen, um über Unternehmensanwendungen auf S3-Daten zuzugreifen.

Diese neue Funktionalität – Unterstützung für die Verwendung des Zugriffs von Endbenutzeridentitäten auf Daten – wird durch die Zuordnung Ihrer S3-Access-Grants-Instance zu einer AWS IAM Identity Center Instance bereitgestellt. IAM Identity Center unterstützt standardbasierte Identitätsanbieter und ist das Zentrum in AWS für alle -Services oder -Funktionen, einschließlich S3 Access Grants, die Endbenutzeridentitäten unterstützen. IAM Identity Center unterstützt über die Funktion für die vertrauenswürdige Weitergabe von Identitäten die Authentifizierung von Unternehmensidentitäten. Weitere Informationen finden Sie unter Vertrauenswürdige Identitätsweitergabe über Anwendungen hinweg.

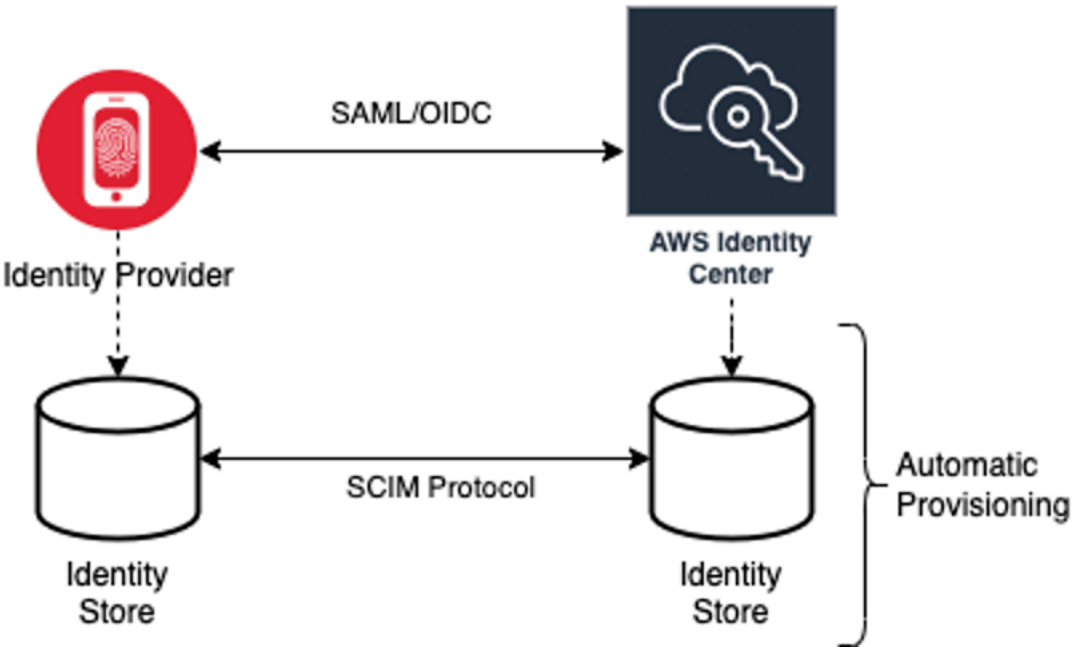
Um mit der Unterstützung von Benutzeridentitäten in S3 Access Grants zu beginnen, müssen Sie zunächst in IAM Identity Center die Identitätsbereitstellung zwischen den Identitätsanbieter Ihres Unternehmens und IAM Identity Center konfigurieren. IAM Identity Center unterstützt Unternehmensidentitätsanbieter wie Okta, Microsoft Entra ID (früherAzure Active Directory) und jeden anderen externen Identitätsanbieter (IdP), der das Protokoll „System for Cross-domain Identity Management (SCIM)“ unterstützt. Wenn Sie IAM Identity Center mit Ihrem IdP verbinden und die automatische Bereitstellung aktivieren, werden die Benutzer und Gruppen im IdP mit dem Identitätsspeicher in IAM Identity Center synchronisiert. Nach diesem Schritt hat IAM Identity Center eine eigene Ansicht Ihrer Benutzer und Gruppen, sodass Sie auf sie verweisen können, indem Sie andere - AWS-Services und -Funktionen wie S3 Access Grants verwenden. Weitere Informationen zur Konfiguration der automatischen Bereitstellung über IAM Identity Center finden Sie unter Automatische Bereitstellung im AWS IAM Identity Center -Benutzerhandbuch.

IAM Identity Center ist in integriert, AWS Organizations sodass Sie Berechtigungen für mehrere zentral verwalten können, AWS-Konten ohne jedes Ihrer Konten manuell zu konfigurieren. In einer typischen Organisation konfiguriert der Identitätsadministrator eine einzelne IAM-Identity-Center-Instance für die gesamte Organisation, die als zentrale Punkt der Identitätssynchronisierung dient. Diese IAM-Identity-Center-Instance wird in der Regel in einer dedizierten AWS-Konto in Ihrer

Organisation ausgeführt. In dieser gemeinsamen Konfiguration können Sie in S3 Access Grants von jedem AWS-Konto in der Organisation auf Benutzer- und Gruppenidentitäten verweisen.

Wenn Ihr AWS Organizations Administrator jedoch noch keine zentrale IAM-Identity-Center-Instance konfiguriert hat, können Sie eine lokale Instance im selben Konto wie Ihre S3-Access-Grants-Instance erstellen. Eine solche Konfiguration ist häufiger für proof-of-concept oder lokale Entwicklungsanwendungsfälle. In allen Fällen muss sich die IAM-Identity-Center-Instance in derselben befinden AWS-Region wie die S3-Access-Grants-Instance, der sie zugeordnet wird.

Im folgenden Diagramm einer IAM-Identity-Center-Konfiguration mit einem externen IdP wurde der IdP mit SCIM konfiguriert, um den Identitätsspeicher des IdP mit dem Identitätsspeicher in IAM Identity Center zu synchronisieren.



Gehen Sie wie folgt vor, um Ihre Unternehmensverzeichnisidentitäten mit S3 Access Grants zu verwenden:

- Richten Sie die automatische Bereitstellung in IAM Identity Center ein, um Benutzer- und Gruppeninformationen Ihres IdP mit IAM Identity Center zu synchronisieren.
- Konfigurieren Sie Ihre externe Identitätsquelle in IAM Identity Center als vertrauenswürdigen Token-Aussteller. Weitere Informationen finden Sie unter Vertrauenswürdige Identitätsweitergabe über Anwendungen hinweg im AWS IAM Identity Center -Benutzerhandbuch.
- Ordnen Sie Ihre S3-Access-Grants-Instance Ihrer IAM-Identity-Center-Instance zu. Sie können dies während der Erstellung Ihrer S3-Access-Grants-Instance durchführen. Wenn Sie Ihre S3-Access-

Grants-Instance bereits erstellt haben, finden Sie weitere Informationen unter Verknüpfen oder Trennen Ihrer IAM-Identity-Center-Instance.

Zugriff auf S3-Daten über Verzeichnisidentitäten

Angenommen, es gibt Unternehmensverzeichnisbenutzer, die über eine Unternehmensanwendung auf Ihre S3-Daten zugreifen müssen, z. B. eine Anwendung zur Dokumentenanzeige, die mit Ihrem externen IdP integriert ist (z. B. Okta), um Benutzer zu authentifizieren. Die Authentifizierung des Benutzers in diesen Anwendungen erfolgt in der Regel über Weiterleitungen im Webbrowser des Benutzers. Da es sich bei den Benutzern im Verzeichnis nicht um IAM-Prinzipale handelt, benötigt Ihre Anwendung IAM-Anmeldeinformationen, um die S3-Access-Grants-API-Operation `GetDataAccess` aufzurufen und im Namen der Benutzer Anmeldeinformationen für den Zugriff auf S3-Daten abzurufen. Im Gegensatz zu IAM-Benutzern und -Rollen, die Anmeldeinformationen selbst erhalten, muss Ihre Anwendung einen Verzeichnisbenutzer darstellen, der keiner IAM-Rolle zugeordnet ist, damit dieser Benutzer über S3 Access Grants Zugriff auf die Daten erhalten kann.

Dieser Wechsel vom authentifizierten Verzeichnisbenutzer zu einem IAM-Aufrufer, der im Namen des Verzeichnisbenutzers Anforderungen an S3 Access Grants senden kann, erfolgt über die IAM-Identity-Center-Funktion für vertrauenswürdige Token-Aussteller. Die Anwendung besitzt nach der Authentifizierung des Verzeichnisbenutzers ein Identitätstoken des IdP (z. B.Okta), das den Verzeichnisbenutzer gemäß Okta darstellt. Über die Konfiguration des vertrauenswürdigen Token-Ausstellers in IAM Identity Center kann die Anwendung dieses Okta-Token (der Okta-Mandant ist als „vertrauenswürdiger Aussteller“ konfiguriert) durch ein anderes Identitätstoken von IAM Identity Center ersetzen, das den Verzeichnisbenutzer in AWS-Services auf sichere Weise darstellt. Die Datenanwendung übernimmt anschließend eine IAM-Rolle und stellt das Token des Verzeichnisbenutzers aus dem IAM Identity Center als zusätzlichen Kontext bereit. Die Anwendung kann die resultierende IAM-Sitzung verwenden, um S3 Access Grants aufzurufen. Das Token stellt sowohl die Identität der Anwendung (den IAM-Prinzipal selbst) als auch die Identität des Verzeichnisbenutzers dar.

Der Hauptschritt dieses Wechsels ist der Token-Austausch. Die Anwendung führt diesen Token-Austausch über den Aufruf der API-Operation `CreateTokenWithIAM` in IAM Identity Center auf. Dies ist natürlich auch ein AWS API-Aufruf und erfordert, dass ein IAM-Prinzipal ihn signiert. Der IAM-Prinzipal, der diese Anforderung sendet, ist in der Regel eine IAM-Rolle, die der Anwendung zugeordnet ist. Wenn die Anwendung beispielsweise in Amazon EC2 ausgeführt wird, wird die Anforderung `CreateTokenWithIAM` in der Regel von der IAM-Rolle ausgeführt, die der EC2-Instance zugeordnet ist, auf der die Anwendung ausgeführt wird. Das Ergebnis eines erfolgreichen `CreateTokenWithIAM` Aufrufs ist ein neues Identitätstoken, das in erkannt wird AWS-Services.

Bevor die Anwendung im Namen des Verzeichnisbenutzers `GetDataAccess` aufrufen kann, muss die Anwendung als Nächstes eine IAM-Sitzung abrufen, die die Identität des Verzeichnisbenutzers enthält. Die Anwendung tut dies mit einer AWS Security Token Service (AWS STS)-`AssumeRole`Anforderung, die auch das IAM-Identity-Center-Token für den Verzeichnisbenutzer als zusätzlichen Identitätskontext enthält. Dieser zusätzliche Kontext ermöglicht IAM Identity Center, die Identität des Verzeichnisbenutzers an den nächsten Schritt weiterzugeben. Die IAM-Rolle, die die Anwendung annimmt, ist die Rolle, die IAM-Berechtigungen benötigt, um die `GetDataAccess`-Operation aufzurufen.

Nach der Übernahme der IAM-Rolle des Identitätsträgers mit dem IAM-Identity-Center-Token des Verzeichnisbenutzers als zusätzlichem Kontext, besitzt die Anwendung nun alle nötigen Komponenten, um im Namen des authentifizierten Verzeichnisbenutzers eine signierte Anforderung an `GetDataAccess` zu senden.

Die Token-Weitergabe basiert auf den folgenden Schritten:

Erstellen einer IAM-Identity-Center-Anwendung

Erstellen Sie zunächst eine neue Anwendung in IAM Identity Center. Diese Anwendung verwendet eine Vorlage, mit der IAM Identity Center feststellen kann, welche Art von Anwendungseinstellungen Sie verwenden können. Für den Befehl zum Erstellen der Anwendung müssen Sie den Amazon Resource Name (ARN) der IAM-Identity-Center-Instance, einen Anwendungsnamen und den Anwendungsanbieter-ARN angeben. Der Anwendungsanbieter ist der SAML- oder OAuth-Anwendungsanbieter, den die Anwendung für Aufrufe an das IAM Identity Center verwendet.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch eigenen Daten:

```
aws sso-admin create-application \
 --instance-arn "arn:aws:sso:::instance/ssoins-ssoins-1234567890abcdef" \
 --application-provider-arn "arn:aws:sso::aws:applicationProvider/custom" \
 --name MyDataApplication
```

Antwort:

```
{
   "ApplicationArn": "arn:aws:sso::123456789012:application/ssoins-
ssoins-1234567890abcdef/apl-abcd1234a1b2c3d"
}
```

Erstellen eines vertrauenswürdigen Token-Ausstellers

Da Sie jetzt Ihre IAM-Identity-Center-Anwendung besitzen, besteht der nächste Schritt in der Konfiguration eines vertrauenswürdigen Token-Ausstellers, der für den Austausch der `IdToken`-Werte Ihres IdP gegen IAM-Identity-Center-Token verwendet wird. In diesem Schritt müssen Sie die folgenden Elemente bereitstellen:

- URL des Identitätsanbieter-Ausstellers
- Name des vertrauenswürdigen Token-Ausstellers
- Pfad zum Anforderungsattribut
- Pfad zum Identitätsspeicherattribut
- JSON Web Key Set (JWKS)-Abrufoption

Der Pfad zum Anforderungsattribut ist das Identitätsanbieterattribut, das für die Zuordnung zum Identitätsspeicherattribut verwendet wird. Normalerweise ist der Pfad zum Anforderungsattribut die E-Mail-Adresse des Benutzers. Sie können jedoch auch andere Attribute für die Zuordnung verwenden.

Erstellen Sie eine Datei mit dem Namen `oidc-configuration.json` und den folgenden Informationen. Wenn Sie diese Datei verwenden möchten, ersetzen Sie *`user input placeholders`* durch eigene Daten.

```
{
  "OidcJwtConfiguration":
     {
      "IssuerUrl": "https://login.microsoftonline.com/a1b2c3d4-abcd-1234-b7d5-
b154440ac123/v2.0",
      "ClaimAttributePath": "preferred_username",
      "IdentityStoreAttributePath": "userName",
      "JwksRetrievalOption": "OPEN_ID_DISCOVERY"
     }
}
```

Führen Sie den folgenden Befehl aus, um den vertrauenswürdigen Token-Aussteller zu erstellen. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws sso-admin create-trusted-token-issuer \
  --instance-arn "arn:aws:sso:::instance/ssoins-1234567890abcdef" \
  --name MyEntraIDTrustedIssuer \
```

```
  --trusted-token-issuer-type OIDC_JWT \
  --trusted-token-issuer-configuration file://./oidc-configuration.json
```

Antwort

```
{
  "TrustedTokenIssuerArn": "arn:aws:sso::123456789012:trustedTokenIssuer/
ssoins-1234567890abcdef/tti-43b4a822-1234-1234-1234-a1b2c3d41234"
}
```

Verbinden der IAM-Identity-Center-Anwendung mit dem vertrauenswürdigen Token-Aussteller

Für den vertrauenswürdigen Token-Aussteller sind einige weitere Konfigurationseinstellungen erforderlich, um zu funktionieren. Legen Sie die Zielgruppe fest, der der vertrauenswürdige Token-Aussteller vertrauen soll. Die Zielgruppe ist der Wert im `IdToken`, der durch den Schlüssel identifiziert wird und in den Einstellungen des Identitätsanbieters zu finden ist. Beispielsweise:

```
1234973b-abcd-1234-abcd-345c5a9c1234
```

Erstellen Sie eine Datei mit dem Namen `grant.json` mit dem folgenden Inhalt. Um diese Datei zu verwenden, ändern Sie die Zielgruppe entsprechend den Einstellungen des Identitätsanbieters und geben den ARN des vertrauenswürdigen Token-Ausstellers an, der vom vorherigen Befehl zurückgegeben wurde.

```
{
   "JwtBearer":
     {
       "AuthorizedTokenIssuers":
         [
           {
             "TrustedTokenIssuerArn": "arn:aws:sso::123456789012:trustedTokenIssuer/
ssoins-1234567890abcdef/tti-43b4a822-1234-1234-1234-a1b2c3d41234",
               "AuthorizedAudiences":
                 [
                   "1234973b-abcd-1234-abcd-345c5a9c1234"
                 ]
            }
         ]
     }
 }
```

Führen Sie den folgenden Beispielbefehl aus. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

```
aws sso-admin put-application-grant \
  --application-arn "arn:aws:sso::123456789012:application/ssoins-
ssoins-1234567890abcdef/apl-abcd1234a1b2c3d" \
  --grant-type "urn:ietf:params:oauth:grant-type:jwt-bearer" \
  --grant file://./grant.json \
```

Mit diesem Befehl werden die Konfigurationseinstellungen des vertrauenswürdigen Token-Ausstellers so festgelegt, dass er der Zielgruppe in der Datei `grant.json` vertraut und diese Zielgruppe mit der Anwendung verknüpft, die im ersten Schritt für den Austausch von Token des Typs `jwt-bearer` erstellt wurde. Die Zeichenfolge `urn:ietf:params:oauth:grant-type:jwt-bearer` ist keine willkürliche Zeichenfolge. Sie ist ein registrierter Namespace in OAuth JSON Web Token (JWT)-Zusicherungsprofilen. Weitere Informationen zu diesem Namespace finden Sie in RFC 7523.

Verwenden Sie als Nächstes den folgenden Befehl, um festzulegen, welche Bereiche der vertrauenswürdige Token-Aussteller beim Austausch von `IdToken`-Werten Ihres Identitätsanbieters einbezieht. Für S3 Access Grants ist der Wert für den Parameter `--scope` `s3:access_grants:read_write`.

```
aws sso-admin put-application-access-scope \
  --application-arn "arn:aws:sso::111122223333:application/ssoins-
ssoins-111122223333abcdef/apl-abcd1234a1b2c3d" \
  --scope "s3:access_grants:read_write"
```

Der letzte Schritt besteht in der Anfügung einer Ressourcenrichtlinie an die IAM-Identity-Center-Anwendung. Diese Richtlinie ermöglicht der IAM-Rolle Ihrer Anwendung das Senden von Anforderungen an die API-Operation `sso-oauth:CreateTokenWithIAM` und den Erhalt von `IdToken`-Werten aus dem IAM Identity Center.

Erstellen Sie eine Datei mit dem Namen `authentication-method.json` mit folgendem Inhalt. Ersetzen Sie *123456789012* durch Ihre Konto-ID.

```
{
   "Iam":
       {
         "ActorPolicy":
             {
```

```
                "Version": "2012-10-17",
                    "Statement":
                    [
                        {
                           "Effect": "Allow",
                            "Principal":
                            {
                              "AWS": "arn:aws:iam::123456789012:role/webapp"
                            },
                           "Action": "sso-oauth:CreateTokenWithIAM",
                            "Resource": "*"
                        }
                    ]
                }
            }
        }
```

Führen Sie zum Anfügen der Richtlinie an die IAM-Identity-Center-Anwendung den folgenden Befehl aus:

```
aws sso-admin put-application-authentication-method \
   --application-arn "arn:aws:sso::123456789012:application/ssoins-
ssoins-1234567890abcdef/apl-abcd1234a1b2c3d" \
   --authentication-method-type IAM \
   --authentication-method file://./authentication-method.json
```

Damit sind die Konfigurationseinstellungen für die Verwendung von S3 Access Grants mit Verzeichnisbenutzern über eine Webanwendung abgeschlossen. Sie können diese Einrichtung direkt in der Anwendung testen oder die API-Operation `CreateTokenWithIAM` aufrufen, indem Sie den folgenden Befehl aus einer zugelassenen IAM-Rolle in der IAM-Identity-Center-Anwendungsrichtlinie verwenden:

```
aws sso-oidc create-token-with-iam \
   --client-id "arn:aws:sso::123456789012:application/ssoins-ssoins-1234567890abcdef/
apl-abcd1234a1b2c3d"  \
   --grant-type urn:ietf:params:oauth:grant-type:jwt-bearer \
   --assertion IdToken
```

Die Antwort wird sieht ungefähr wie folgt aus:

```
{
```

```
    "accessToken": "<suppressed long string to reduce space>",
    "tokenType": "Bearer",
    "expiresIn": 3600,
    "refreshToken": "<suppressed long string to reduce space>",
    "idToken": "<suppressed long string to reduce space>",
    "issuedTokenType": "urn:ietf:params:oauth:token-type:refresh_token",
    "scope": [
      "sts:identity_context",
      "s3:access_grants:read_write",
      "openid",
      "aws"
    ]
}
```

Wenn Sie den mit base64 kodierten `IdToken`-Wert dekodieren, können Sie die Schlüssel-Wert-Paare im JSON-Format anzeigen. Der Schlüssel `sts:identity_context` enthält den Wert, den Ihre Anwendung in der `sts:AssumeRole`-Anforderung senden muss, um die Identitätsinformationen des Verzeichnisbenutzers einzuschließen. Dies ist ein Beispiel für das dekodierte `IdToken`:

```
{
    "aws:identity_store_id": "d-996773e796",
    "sts:identity_context": "AQoJb3JpZ2luX2VjEOTtl;<SUPRESSED>",
    "sub": "83d43802-00b1-7054-db02-f1d683aacba5",
    "aws:instance_account": "123456789012",
    "iss": "https://identitycenter.amazonaws.com/ssoins-1234567890abcdef",
    "sts:audit_context": "AQoJb3JpZ2luX2VjEOT<SUPRESSED>==",
    "aws:identity_store_arn": "arn:aws:identitystore::232642235904:identitystore/
d-996773e796",
    "aud": "abcd12344U0gi7n4Yyp0-WV1LWNlbnRyYWwtMQ",
    "aws:instance_arn": "arn:aws:sso:::instance/ssoins-6987d7fb04cf7a51",
    "aws:credential_id": "EXAMPLEHI5glPh40y9TpApJn8...",
    "act": {
       "sub": "arn:aws:sso::232642235904:trustedTokenIssuer/
ssoins-6987d7fb04cf7a51/43b4a822-1020-7053-3631-cb2d3e28d10e"
    },
    "auth_time": "2023-11-01T20:24:28Z",
    "exp": 1698873868,
    "iat": 1698870268
}
```

Sie können den Wert aus `sts:identity_context` abrufen und diese Informationen in einem `sts:AssumeRole`-Aufruf weitergeben. Dies ist ein CLI-Beispiel für die Syntax. Bei der Rolle, die

übernommen werden soll, handelt es sich um eine temporäre Rolle mit Berechtigungen für den Aufruf von `s3:GetDataAccess`.

```
aws sts assume-role \
   --role-arn "arn:aws:iam::123456789012:role/temp-role" \
   --role-session-name "TempDirectoryUserRole" \
   --provided-contexts ProviderArn="arn:aws:iam::aws:contextProvider/
IdentityCenter",ContextAssertion="value from sts:identity_context"
```

Sie können nun die bei diesem Aufruf erhaltenen Anmeldeinformationen verwenden, um die API-Operation `s3:GetDataAccess` aufzurufen und die endgültigen Anmeldeinformationen für den Zugriff auf Ihre S3-Ressourcen zu erhalten.

Erste Schritte mit S3 Access Grants

Amazon S3 Access Grants ist eine Amazon-S3-Funktion, die eine skalierbare Lösung für die Zugriffssteuerung für Ihre S3-Daten bereitstellt. S3 Access Grants ist ein Anbieter von S3-Anmeldeinformationen. Das bedeutet, dass Sie Ihre Liste von Gewährungen und deren Zugriffsebenen bei S3 Access Grants registrieren. Wenn Benutzer oder Clients anschließend Zugriff auf Ihre S3-Daten benötigen, fordern sie zunächst Anmeldeinformationen von S3 Access Grants an. Wenn es eine entsprechende Gewährung gibt, die den Zugriff autorisiert, stellt S3 Access Grants temporäre Zugangsdaten mit geringsten Berechtigungen bereit. Die Benutzer oder Clients können dann mit den von S3 Access Grants bereitgestellten Anmeldeinformationen auf Ihre S3-Daten zuzugreifen. Wenn Ihre S3-Datenanforderungen eine komplexe oder umfangreiche Berechtigungskonfiguration erfordern, können Sie mit S3 Access Grants die S3-Datenberechtigungen für Benutzer, Gruppen, Rollen und Anwendungen skalieren.

Für die meisten Anwendungsfälle können Sie die Zugriffskontrolle für Ihre S3-Daten mithilfe von AWS Identity and Access Management (IAM) mit Bucket-Richtlinien oder identitätsbasierten IAM-Richtlinien verwalten.

Wenn es jedoch komplexe Anforderungen an die S3-Zugriffssteuerung gibt, z. B. die folgenden, könnten Sie erheblich von S3 Access Grants profitieren:

- Sie erreichen das Größenlimit von 20 KB für die Bucket-Richtlinie.
- Sie gewähren menschlichen Identitäten, z. B. Benutzern und Gruppen in Microsoft Entra ID (früher Azure Active Directory), Okta oder Ping, Zugriff auf S3-Daten für Analyse- und Big-Data-Anwendungen.

- Sie müssen einen kontoübergreifenden Zugriff gewähren, ohne die IAM-Richtlinien häufig aktualisieren zu müssen.
- Ihre Daten sind unstrukturiert und befinden sich eher auf Objektebene, statt ein strukturiertes Zeilen- und Spaltenformat aufzuweisen.

Der Workflow in S3 Access Grants ist wie folgt:

Schritte	Beschreibung
1	Erstellen einer S3-Access-Grants-Instance Initiieren Sie zunächst eine S3-Access-Grants-Instance, die Ihre individuellen Zugriffsberechtigungen enthält.
2	Registrieren eines Speicherorts Registrieren Sie als Nächstes einen S3-Datenspeicherort (z. B. den Standardspeicherort `s3://`). Geben Sie dann eine IAM-Standardrolle an, die S3 Access Grants bei der Bereitstellung des Zugriffs auf den S3-Datenspeicherort übernimmt. Sie können bestimmten Buckets oder Präfixen auch benutzerdefinierte Speicherorte hinzufügen und diese zu benutzerdefinierten IAM-Rollen zuordnen.
3	Erstellen von Gewährungen Erstellen Sie individuelle Berechtigungsgewährungen. Geben Sie in diesen Berechtigungsgewährungen den registrierten S3-Speicherort, den Umfang des Datenzugriffs innerhalb des Speicherorts, die Identität des Gewährungsempfängers und die Zugriffsebene (`READ`, `WRITE` oder `READWRITE`) an.
4	Anfordern von Zugriff auf S3-Daten Wenn Benutzer, Anwendungen und auf S3-Daten zugreifen AWS-Services möchten, stellen sie zunächst eine Zugriffsanforderung. S3 Access Grants legt fest, ob die Anforderung autorisiert werden soll. Wenn es eine entsprechende

Schritte	Beschreibung
	Gewährung gibt, die den Zugriff autorisiert, verwendet S3 Access Grants die IAM-Rolle des registrierten Speicherorts, der mit dieser Gewährung verknüpft ist, um temporäre Anmeldeinformationen an den Anforderer zurückzugeben.
5	Zugriff auf S3-Daten Anwendungen verwenden für den Zugriff auf S3-Daten die temporären Anmeldeinformationen, die von S3 Access Grants bereitgestellt wurden.

Erstellen einer S3-Access-Grants-Instance

Um mit der Verwendung von Amazon S3 Access Grants zu beginnen, erstellen Sie zunächst eine S3-Access-Grants-Instance. Sie können nur eine S3-Access-Grants-Instance pro AWS-Region und Konto erstellen. Die S3-Access-Grants-Instance dient als Container für Ihre S3-Access-Grants-Ressourcen, zu denen registrierte Speicherorte und Gewährungen gehören.

Mit S3 Access Grants können Sie Berechtigungserteilungen für Ihre S3-Daten für AWS Identity and Access Management (IAM)-Benutzer und -Rollen erstellen. Wenn Sie Ihr Unternehmensidentitätsverzeichnis zu hinzugefügt haben AWS IAM Identity Center, können Sie diese IAM-Identity-Center-Instance Ihres Unternehmensverzeichnisses Ihrer S3-Access-Grants-Instance zuordnen. Anschließend können Sie Zugriffsgewährungen für die Benutzer und Gruppen Ihres Unternehmens erstellen. Wenn Sie Ihr Unternehmensverzeichnis noch nicht zu IAM Identity Center hinzugefügt haben, können Sie Ihre S3-Access-Grants-Instance auch noch später mit einer IAM-Identity-Center-Instance verknüpfen.

Sie können eine S3-Access-Grants-Instance mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und AWS SDKs erstellen.

Verwenden der S3-Konsole

Bevor Sie mit S3 Access Grants Zugriff auf Ihre S3-Daten gewähren können, müssen Sie zunächst eine S3-Access-Grants-Instance in derselben AWS-Region wie Ihre S3-Daten erstellen.

Voraussetzungen

Wenn Sie mittels Identitäten aus Ihrem Unternehmensverzeichnis Zugriff auf Ihre S3-Daten gewähren möchten, müssen Sie Ihr Unternehmensidentitätsverzeichnis zu AWS IAM Identity Center hinzufügen. Wenn Sie dazu noch nicht bereit sind, können Sie Ihre S3-Access-Grants-Instance auch später noch mit einer IAM-Identity-Center-Instance verknüpfen.

So erstellen Sie eine S3-Access-Grants-Instance

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Option S3-Access-Grants-Instance erstellen aus.

   a. Überprüfen Sie in Schritt 1 des Assistenten Access-Grants-Instance einrichten, ob Sie die Instance in der aktuellen AWS-Region erstellen. Stellen Sie sicher, dass dies derselbe ist AWS-Region , in dem sich Ihre S3-Daten befinden. Sie können eine S3-Access-Grants-Instance pro AWS-Region und Konto erstellen.

   b. (Optional) Wenn Sie Ihr Unternehmensidentitätsverzeichnis zu hinzugefügt haben AWS IAM Identity Center, können Sie diese IAM-Identity-Center-Instance Ihres Unternehmensverzeichnisses Ihrer S3-Access-Grants-Instance zuordnen.

      Wählen Sie hierzu IAM-Identity-Center-Instance in ***Region*** hinzufügen aus. Geben Sie dann den Amazon-Ressourcennamen (ARN) der IAM-Identity-Center-Instance ein.

      Wenn Sie Ihr Unternehmensverzeichnis noch nicht zu IAM Identity Center hinzugefügt haben, können Sie Ihre S3-Access-Grants-Instance auch noch später mit einer IAM-Identity-Center-Instance verknüpfen.

   c. Um die S3-Access-Grants-Instance zu erstellen, wählen Sie Weiter aus. Informationen zur Registrierung eines Speicherorts finden Sie unter Schritt 2 – Registrieren eines Speicherorts.
4. Wenn Weiter oder S3-Access-Grants-Instance erstellen deaktiviert ist:

   Instance ksnn nicht erstellt werden

   - Möglicherweise gibt es bereits eine S3-Access-Grants-Instance in derselben AWS-Region. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus. Scrollen Sie auf der Seite S3 Access Grants nach unten zum Abschnitt S3-Access-Grants-Instance in Ihrem Konto, um festzustellen, ob es bereits eine Instance gibt.
   - Möglicherweise verfügen Sie nicht über die Berechtigung `s3:CreateAccessGrantsInstance`, die für die Erstellung einer S3-Access-Grants-

Instance erforderlich ist. Nehmen Sie Kontakt mit Ihrem Kontoadministrator auf. Informationen zu zusätzlichen Berechtigungen, die für die Zuordnung einer IAM-Identity-Center-Instance zu Ihrer S3-Access-Grants-Instance erforderlich sind, finden Sie unter CreateAccessGrantsInstance.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen.

Example Erstellen einer S3-Access-Grants-Instance

```
aws s3control create-access-grants-instance \
--account-id 111122223333 \
--region us-east-2
```

Antwort:

```
{
    "CreatedAt": "2023-05-31T17:54:07.893000+00:00",
    "AccessGrantsInstanceId": "default",
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default"
}
```

Verwenden der REST-API

Sie können die Amazon-S3-REST-API verwenden, um eine S3-Access-Grants-Instance zu erstellen. Informationen zur REST-API-Unterstützung für die Verwaltung einer S3-Access-Grants-Instance finden Sie in den folgenden Abschnitten in der Amazon-Simple-Storage-Service-API-Referenz:

- AssociateAccessGrantsIdentityCenter
- CreateAccessGrantsInstance
- DeleteAccessGrantsInstance
- DissociateAccessGrantsIdentityCenter

- GetAccessGrantsInstance
- GetAccessGrantsInstanceForPrefix
- GetAccessGrantsInstanceResourcePolicy
- ListAccessGrantsInstances
- PutAccessGrantsInstanceResourcePolicy

Verwenden der AWS SDKs

Dieser Abschnitt enthält ein Beispiel dafür, wie Sie mit den AWS SDKs eine S3-Access-Grants-Instance erstellen.

Java

In diesem Beispiel wird die S3-Access-Grants-Instance erstellt, die als Container für Ihre individuellen Zugriffsgewährungen dient. Sie können eine S3-Access-Grants-Instance pro AWS-Region in Ihrem Konto haben. Die Antwort enthält die Instance-ID `default` und den Amazon-Ressourcennamen (ARN), der für Ihre S3-Access-Grants-Instance generiert wurde.

Example Erstellen einer S3-Access-Grants-Instance-Anforderung

```
public void createAccessGrantsInstance() {
CreateAccessGrantsInstanceRequest createRequest =
 CreateAccessGrantsInstanceRequest.builder().accountId("111122223333").build();
CreateAccessGrantsInstanceResponse createResponse =
 s3Control.createAccessGrantsInstance(createRequest);LOGGER.info("CreateAccessGrantsInstance
 " + createResponse);
}
```

Antwort:

```
CreateAccessGrantsInstanceResponse(
CreatedAt=2023-06-07T01:46:20.507Z,
AccessGrantsInstanceId=default,
AccessGrantsInstanceArn=arn:aws:s3:us-east-2:111122223333:access-grants/default)
```

Themen

- Anzeigen der Details einer S3-Access-Grants-Instance

- Verknüpfen oder Trennen Ihrer IAM-Identity-Center-Instance
- Löschen einer S3-Access-Grants-Instance

Anzeigen der Details einer S3-Access-Grants-Instance

Sie können Details zu Ihrer Amazon-S3-Access-Grants-Instance in einer bestimmten AWS-Region anzeigen. Sie können auch Ihre S3-Access-Grants-Instances auflisten, einschließlich der Instances, die über AWS Resource Access Manager () für Sie freigegeben wurden AWS RAM.

Sie können die Details Ihrer S3-Access-Grants-Instance anzeigen oder Ihre S3-Access-Grants-Instances mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und der AWS SDKs auflisten.

Verwenden der S3-Konsole

So zeigen Sie eine S3-Access-Grants-Instance an

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.
4. Auf der Seite S3 Access Grants werden Ihre S3-Access-Grants-Instances und alle kontoübergreifenden Instances aufgeführt, die mit Ihrem Konto geteilt wurden. Um die Details einer Instance anzuzeigen, wählen Sie Details anzeigen aus.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen.

Example – Anzeigen der Details einer S3-Access-Grants-Instance

```
aws s3control get-access-grants-instance \
 --account-id 111122223333 \
```

```
 --region us-east-2
```

Antwort:

```
{
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2: 111122223333:access-grants/
default",
    "AccessGrantsInstanceId": "default",
    "CreatedAt": "2023-05-31T17:54:07.893000+00:00"
}
```

Example – Auflisten aller S3-Access-Grants-Instances für ein Konto

Diese Aktion listet die S3-Access-Grants-Instances für ein Konto auf. Sie können nur eine S3-Access-Grants-Instance pro haben AWS-Region. Diese Aktion listet auch weitere kontoübergreifende S3-Access-Grants-Instances auf, auf die Ihr Konto Zugriff hat.

```
aws s3control list-access-grants-instances \
 --account-id 111122223333 \
 --region us-east-2
```

Antwort:

```
{
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2: 111122223333:access-grants/
default",
    "AccessGrantsInstanceId": "default",
    "CreatedAt": "2023-05-31T17:54:07.893000+00:00"
}
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für die Verwaltung einer S3-Access-Grants-Instance finden Sie in den folgenden Abschnitten in der Amazon-Simple-Storage-Service-API-Referenz:

- GetAccessGrantsInstance
- GetAccessGrantsInstanceForPrefix
- ListAccessGrantsInstances

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele dafür, wie Sie die Details einer S3-Access-Grants-Instance mithilfe der - AWS SDKs abrufen.

Um die folgenden Beispiele zu verwenden, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen.

Java

Example – Abrufen einer S3-Access-Grants-Instance

```
public void getAccessGrantsInstance() {
GetAccessGrantsInstanceRequest getRequest = GetAccessGrantsInstanceRequest.builder()
.accountId("111122223333")
.build();
GetAccessGrantsInstanceResponse getResponse =
 s3Control.getAccessGrantsInstance(getRequest);
LOGGER.info("GetAccessGrantsInstanceResponse: " + getResponse);
}
```

Antwort:

```
GetAccessGrantsInstanceResponse(
AccessGrantsInstanceArn=arn:aws:s3:us-east-2: 111122223333:access-grants/default,
CreatedAt=2023-06-07T01:46:20.507Z)
```

Example – Auflisten aller S3-Access-Grants-Instances für ein Konto

Diese Aktion listet die S3-Access-Grants-Instances für ein Konto auf. Es kann pro Region nur eine S3-Access-Grants-Instance geben. Diese Aktion kann auch weitere kontoübergreifende S3-Access-Grants-Instances auflisten, auf die Ihr Konto Zugriff hat.

```
public void listAccessGrantsInstances() {
ListAccessGrantsInstancesRequest listRequest =
 ListAccessGrantsInstancesRequest.builder()
.accountId("111122223333")
.build();
ListAccessGrantsInstancesResponse listResponse =
 s3Control.listAccessGrantsInstances(listRequest);
LOGGER.info("ListAccessGrantsInstancesResponse: " + listResponse);
}
```

Antwort:

```
ListAccessGrantsInstancesResponse(
AccessGrantsInstancesList=[
ListAccessGrantsInstanceEntry(
AccessGrantsInstanceId=default,
AccessGrantsInstanceArn=arn:aws:s3:us-east-2:111122223333:access-grants/default,
CreatedAt=2023-06-07T04:28:11.728Z
)
]
)
```

Verknüpfen oder Trennen Ihrer IAM-Identity-Center-Instance

In Amazon S3 Access Grants können Sie die AWS IAM Identity Center Instance Ihres Unternehmensidentitätsverzeichnisses einer S3-Access-Grants-Instance zuordnen. Danach können Sie zusätzlich zu AWS Identity and Access Management (IAM)-Benutzern und -Rollen Zugriffsgewährungen für Ihre Benutzer und Gruppen im Unternehmensverzeichnis erstellen.

Wenn Sie nicht weiter Zugriffsgewährungen für die Benutzer und Gruppen in Ihrem Unternehmensverzeichnis erstellen möchten, können Sie Ihre IAM-Identity-Center-Instance von Ihrer S3-Access-Grants-Instance trennen.

Sie können eine IAM-Identity-Center-Instance über die Amazon-S3-Konsole, die AWS Command Line Interface (AWS CLI), die Amazon-S3-REST-API und die AWS -SDKs verknüpfen oder trennen.

Verwenden der S3-Konsole

Bevor Sie Ihre IAM-Identity-Center-Instance mit Ihrer S3-Access-Grants-Instance verknüpfen können, müssen Sie Ihr Unternehmensidentitätsverzeichnis dem IAM Identity Center hinzufügen. Weitere Informationen finden Sie unter the section called “S3 Access Grants und Unternehmensverzeichnisidentitäten”.

So verknüpfen Sie eine IAM-Identity-Center-Instance mit einer S3-Access-Grants-Instance

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.

4. Wählen Sie Details anzeigen für die Instance aus.
5. Wählen Sie auf der Detailseite im Abschnitt IAM Identity Center die Hinzufügung einer IAM-Identity-Center-Instance oder die Aufhebung der Registrierung einer bereits verknüpften IAM-Identity-Center-Instance aus.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

Example – Verknüpfen einer IAM-Identity-Center-Instance mit einer S3-Access-Grants-Instance

```
aws s3control associate-access-grants-identity-center \
 --account-id 111122223333 \
 --identity-center-arn arn:aws:sso:::instance/ssoins-1234a567bb89012c \
 --profile access-grants-profile \
 --region eu-central-1

// No response body
```

Example – Trennen einer IAM-Identity-Center-Instance von einer S3-Access-Grants-Instance

```
aws s3control dissociate-access-grants-identity-center \
 --account-id 111122223333 \
 --profile access-grants-profile \
 --region eu-central-1

// No response body
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für die Verwaltung der Verknüpfung zwischen einer IAM-Identity-Center-Instance und einer S3-Access-Grants-Instance finden Sie in den folgenden Abschnitten in der Amazon-Simple-Storage-Service-API-Referenz:

- AssociateAccessGrantsIdentityCenter
- DissociateAccessGrantsIdentityCenter

Löschen einer S3-Access-Grants-Instance

Sie können eine Amazon S3-Access-Grants-Instance aus einem AWS-Region in Ihrem Konto löschen. Bevor Sie jedoch eine S3-Access-Grants-Instance löschen können, müssen Sie Folgendes ausführen:

- Löschen Sie alle Ressourcen innerhalb der S3-Access-Grants-Instance, einschließlich aller Gewährungen und Speicherorte. Weitere Informationen finden Sie unter Eine Gewährung löschen und Einen Speicherort löschen.
- Wenn Sie Ihrer S3-Access-Grants- AWS IAM Identity Center Instance eine Instance zugeordnet haben, müssen Sie die Zuordnung der IAM-Identity-Center-Instance aufheben. Weitere Informationen finden Sie unter Verknüpfen oder Trennen Ihrer IAM-Identity-Center-Instance.

> **Important**
>
> Wenn Sie eine S3-Access-Grants-Instance löschen, wird sie dauerhaft gelöscht. Dieser Vorgang kann nicht rückgängig gemacht werden. Alle Gewährungsempfänger, die über die Gewährungen in dieser S3-Access-Grants-Instance Zugriff erhalten haben, verlieren den Zugriff auf Ihre S3-Daten.

Sie können eine S3-Access-Grants-Instance mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und der AWS SDKs löschen.

Verwenden der S3-Konsole

So löschen Sie eine S3-Access-Grants-Instance

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.
4. Wählen Sie Details anzeigen für die Instance aus.
5. Wählen Sie oben rechts auf der Instance-Detailseite Instance löschen aus.
6. Wählen Sie im anschließend angezeigten Dialogfeld Löschen aus. Diese Aktion kann nicht mehr rückgängig gemacht werden.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

> **Note**
>
> Bevor Sie eine S3-Access-Grants-Instance löschen können, müssen Sie zunächst alle Gewährungen und Speicherorte löschen, die in der S3-Access-Grants-Instance erstellt wurden. Wenn Sie eine IAM-Identity-Center-Instance mit Ihrer S3-Access-Grants-Instance verknüpft haben, müssen Sie diese zuerst trennen.

Example – Löschen einer S3-Access-Grants-Instance

```
aws s3control delete-access-grants-instance \
--account-id 111122223333 \
--profile access-grants-profile \
--region us-east-2 \
--endpoint-url https://s3-control.us-east-2.amazonaws.com \

 // No response body
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für das Löschen einer S3-Access-Grants-Instance finden Sie unter DeleteAccessGrantsInstance in der Amazon-Simple-Storage-Service-API-Referenz.

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele dafür, wie Sie über die AWS SDKs eine S3-Access-Grants-Instance löschen können.

Wenn Sie das folgende Beispiel verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen.

Java

> **Note**
>
> Bevor Sie eine S3-Access-Grants-Instance löschen können, müssen Sie zunächst alle Gewährungen und Speicherorte löschen, die in der S3-Access-Grants-Instance erstellt wurden. Wenn Sie eine IAM-Identity-Center-Instance mit Ihrer S3-Access-Grants-Instance verknüpft haben, müssen Sie diese zuerst trennen.

Example – Löschen einer S3-Access-Grants-Instance

```
public void deleteAccessGrantsInstance() {
DeleteAccessGrantsInstanceRequest deleteRequest =
 DeleteAccessGrantsInstanceRequest.builder()
.accountId("111122223333")
.build();
DeleteAccessGrantsInstanceResponse deleteResponse =
 s3Control.deleteAccessGrantsInstance(deleteRequest);
LOGGER.info("DeleteAccessGrantsInstanceResponse: " + deleteResponse);
}
```

Registrieren eines Speicherorts

Nachdem Sie eine Amazon S3-Access-Grants-Instance in einem in Ihrem Konto erstellt haben, können Sie einen S3-Speicherort in dieser Instance registrieren. AWS-Region Ein Speicherort ist eine S3-Ressource, die Daten enthält, auf die Sie Zugriff gewähren möchten. Sie können den Standardspeicherort registrieren, `s3://`, bei dem es sich um alle Ihre Buckets in der handelt AWS-Region, und dann den Zugriffsumfang später einschränken, wenn Sie individuelle Zugriffsgewährungen erstellen. Sie können auch einen bestimmten Bucket oder einen Bucket mit Präfix als Speicherort registrieren.

Sie müssen zunächst mindestens einen Speicherort bei Ihrer S3-Access-Grants-Instance registrieren, bevor Sie Zugriffsgewährungen erstellen können. Wenn Sie einen Speicherort registrieren, müssen Sie auch die AWS Identity and Access Management (IAM)-Rolle angeben, die S3 Access Grants übernimmt, um Speicherort-Laufzeitanforderungen zu erfüllen und den Umfang der Berechtigungen zur Laufzeit auf die spezifische Gewährung einzuschränken.

S3-URI	IAM-Rolle	Beschreibung
`s3://`	*`Default-IAM-role`*	Der Standardspeicherort (`s3://`) enthält alle Buckets in der AWS-Region.
`s3://`*`DOC-EXAMPLE-BUCKET1`*`/`	*`IAM-role-For-bucket`*	Dieser Speicherort enthält alle Objekte im angegebenen Bucket.

Bevor Sie einen Speicherort registrieren können, muss die folgende Voraussetzung erfüllt sein:

- Erstellen Sie einen oder mehrere Buckets mit den Daten, auf die Sie Zugriff gewähren möchten. Diese Buckets müssen sich in derselben AWS-Region wie Ihre S3-Access-Grants-Instance befinden. Weitere Informationen finden Sie unter Bucket erstellen.

  Informationen zum Hinzufügen eines Präfixes zu einem Bucket finden Sie unter Erstellen von Objektschlüsselnamen.
- Erstellen Sie eine IAM-Rolle und gewähren Sie dem S3-Access-Grants-Service-Prinzipal in der Ressourcenrichtliniendatei Zugriff auf diese Rolle. Hierzu können Sie eine JSON-Datei mit den folgenden Anweisungen erstellen. Informationen zum Hinzufügen der Ressourcenrichtlinie zu Ihrem Konto finden Sie unter Erstellen und Anfügen Ihrer ersten vom Kunden verwalteten Richtlinie.

  TestRolePolicy.json

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1234567891011",
      "Action": ["sts:AssumeRole", "sts:SetSourceIdentity", "sts:SetContext"],
      "Effect": "Allow",
      "Principal": {"Service":"access-grants.s3.amazonaws.com"}
    }
  ]
}
```

- Erstellen Sie eine IAM-Richtlinie, um Amazon-S3-Berechtigungen zur IAM-Rolle hinzuzufügen. Schauen Sie sich die folgende Beispieldatei `iam-policy.json` an und ersetzen Sie die *`user input placeholders`* durch eigene Daten.

> **Note**
>
> Wenn Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln verwenden, um Ihre Daten zu verschlüsseln, fügt das folgende Beispiel der Richtlinie die erforderlichen AWS KMS Berechtigungen für die IAM-Rolle hinzu. Wenn Sie diese Funktion nicht verwenden, können Sie diese Berechtigungen aus Ihrer IAM-Richtlinie entfernen.
>
> Damit die IAM-Rolle nur für den Zugriff auf Daten in S3 verwendet werden kann, wenn die Anmeldeinformationen über S3 Access Grants verteilt werden, fügen Sie eine `Condition`-Anweisung hinzu, die die S3-Access-Grants-Instance (`s3:AccessGrantsInstance:`*`InstanceArn`*) in Ihrer IAM-Richtlinie angibt, wie in diesem Beispiel gezeigt.

iam-policy.json

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
         "Sid": "ObjectLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:GetObjectAcl",
            "s3:GetObjectVersionAcl",
            "s3:ListMultipartUploadParts"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
```

```
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:region:accountId:access-
grants/instanceId"]
            }
        }
      },
      {
         "Sid": "ObjectLevelWritePermissions",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:PutObjectVersionAcl",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion",
            "s3:AbortMultipartUpload"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:AWS-
Region:accountId:access-grants/instanceId"]
            }
         }
      },
      {
         "Sid": "BucketLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:AWS-
Region:accountId:access-grants/instanceId"]
            }
         }
      },
```

```
      {
         "Sid": "KMSPermissions",
         "Effect":"Allow",
         "Action":[
            "kms:Decrypt",
            "kms:GenerateDataKey"
         ],
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Sie können einen Speicherort in Ihrer S3-Access-Grants-Instance mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API oder der AWS SDKs registrieren.

Verwenden der S3-Konsole

Bevor Sie mit S3 Access Grants Zugriff auf Ihre S3-Daten gewähren können, müssen Sie über mindestens einen registrierten Speicherort verfügen.

So registrieren Sie einen Speicherort in Ihrer S3-Access-Grants-Instance

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.

   Wenn Sie die S3-Access-Grants-Instance zum ersten Mal verwenden, müssen Sie Schritt 1 – Erstellen einer S3-Access-Grants-Instance abgeschlossen haben und zu Schritt 2 des Assistenten Access-Grants-Instance einrichten gewechselt sein. Wenn Sie bereits über eine S3-Access-Grants-Instance verfügen, wählen Sie Details anzeigen und dann auf der Registerkarte Speicherorte die Option Speicherort registrieren aus.

   a. Wählen Sie in Speicherortumfang die Option S3 durchsuchen aus oder geben Sie den S3-URI-Pfad zu dem Standort ein, den Sie registrieren möchten. Informationen zu den S3-URI-

Formaten finden Sie in der Tabelle Speicherortformate. Nach der Eingabe eines URI können Sie Anzeigen auswählen, um den Speicherort zu durchsuchen.

b. Wählen Sie in IAM-Rolle eine der folgenden Optionen aus:

- Auswahl aus vorhandenen IAM-Rollen

  Wählen Sie in der Dropdown-Liste eine IAM-Rolle aus. Wählen Sie nach der Auswahl einer Rolle Anzeigen aus, um sicherzustellen, dass diese Rolle die notwendigen Berechtigungen für die Verwaltung des Speicherorts besitzt, den Sie registrieren. Stellen Sie insbesondere sicher, dass diese Rolle S3 Access Grants die Berechtigungen `sts:AssumeRole` und `sts:SetSourceIdentity` gewährt.

- Eingabe des ARN der IAM-Rolle

  Navigieren Sie zur IAM-Konsole . Kopieren Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle und fügen Sie ihn in dieses Feld ein.

c. Wählen Sie Weiter oder Speicherort registrieren aus, um dies abzuschließen.

4. Fehlerbehebung

   Speicherort kann nicht registriert werden

   - Der Speicherort ist möglicherweise bereits registriert.

     Möglicherweise besitzen Sie nicht die Berechtigung `s3:CreateAccessGrantsLocation` für die Registrierung von Speicherorten. Nehmen Sie Kontakt mit Ihrem Kontoadministrator auf.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Sie können in Ihrer S3-Access-Grants-Instance den Standardspeicherort (`s3://`) oder einen benutzerdefinierten Speicherort registrieren. Sie müssen zunächst eine IAM-Rolle mit Prinzipalzugriff auf den Speicherort erstellen und dann S3 Access Grants die Berechtigung gewähren, diese Rolle anzunehmen.

Um die folgenden Beispielbefehle zu verwenden, ersetzen Sie *`user input placeholders`* durch eigene Daten.

Example Erstellen einer Ressourcenrichtlinie

Erstellen Sie eine Richtlinie, die S3 Access Grants die Übernahme der IAM-Rolle ermöglicht. Hierzu können Sie eine JSON-Datei mit den folgenden Anweisungen erstellen. Informationen zum Hinzufügen der Ressourcenrichtlinie zu Ihrem Konto finden Sie unter Erstellen und Anfügen Ihrer ersten vom Kunden verwalteten Richtlinie.

TestRolePolicy.json

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1234567891011",
      "Action": ["sts:AssumeRole", "sts:SetSourceIdentity"],
      "Effect": "Allow",
      "Principal": {"Service":"access-grants.s3.amazonaws.com"}
    }
  ]
}
```

Example Erstellen der Rolle

Führen Sie den folgenden IAM-Befehl aus, um die IAM-Rolle zu erstellen.

```
aws iam create-role --role-name accessGrantsTestRole \
 --region us-east-2 \
 --assume-role-policy-document file://TestRolePolicy.json
```

Wenn Sie den Befehl `create-role` ausführen, wird die Richtlinie zurückgegeben:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "accessGrantsTestRole",
        "RoleId": "AROASRDGX4WM4GH55GIDA",
        "Arn": "arn:aws:iam::111122223333:role/accessGrantsTestRole",
        "CreateDate": "2023-05-31T18:11:06+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
```

```
                    "Sid": "Stmt1685556427189",
                    "Action": [
                        "sts:AssumeRole",
                        "sts:SetSourceIdentity"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service":"access-grants.s3.amazonaws.com"
                    }
                }
            ]
        }
    }
}
```

Example

Erstellen Sie eine IAM-Richtlinie, um Amazon-S3-Berechtigungen zur IAM-Rolle hinzuzufügen. Schauen Sie sich die folgende Beispieldatei `iam-policy.json` an und ersetzen Sie die *`user input placeholders`* durch eigene Daten.

> **Note**
>
> Wenn Sie die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln verwenden, um Ihre Daten zu verschlüsseln, fügt das folgende Beispiel der Richtlinie die erforderlichen AWS KMS Berechtigungen für die IAM-Rolle hinzu. Wenn Sie diese Funktion nicht verwenden, können Sie diese Berechtigungen aus Ihrer IAM-Richtlinie entfernen.
>
> Damit die IAM-Rolle nur für den Zugriff auf Daten in S3 verwendet werden kann, wenn die Anmeldeinformationen über S3 Access Grants verteilt werden, fügen Sie eine `Condition`-Anweisung hinzu, die die S3-Access-Grants-Instance (`s3:AccessGrantsInstance:`*`InstanceArn`*) in Ihrer IAM-Richtlinie angibt, wie in diesem Beispiel gezeigt. Wenn Sie das folgende Beispiel verwenden, ersetzen Sie die *`user input placeholders`* durch eigene Daten.

iam-policy.json

```
{
```

```
   "Version":"2012-10-17",
   "Statement": [
       {
         "Sid": "ObjectLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:GetObjectAcl",
            "s3:GetObjectVersionAcl",
            "s3:ListMultipartUploadParts"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:region:accountId:access-
grants/instanceId"]
            }
        }
      },
      {
         "Sid": "ObjectLevelWritePermissions",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:PutObjectVersionAcl",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion",
            "s3:AbortMultipartUpload"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:AWS-Region:accountId:access-
grants/instanceId"]
            }
         }
```

```
      },
      {
         "Sid": "BucketLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:AWS-Region:accountId:access-
grants/instanceId"]
            }
         }
      },
      {
         "Sid": "KMSPermissions",
         "Effect":"Allow",
         "Action":[
            "kms:Decrypt",
            "kms:GenerateDataKey"
         ],
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Example

Führen Sie den folgenden Befehl aus:

```
aws iam put-role-policy \
--role-name accessGrantsTestRole \
--policy-name accessGrantsTestRole \
--policy-document file://iam-policy.json
```

Example Registrieren des Standardspeicherorts

```
aws s3control create-access-grants-location \
 --account-id 111122223333 \
 --location-scope s3:// \
 --iam-role-arn arn:aws:iam::111122223333:role/accessGrantsTestRole
```

Antwort:

```
{"CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "default",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/default",
    "LocationScope": "s3://”
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
}
```

Example Registrieren eines benutzerdefinierten Speicherorts

```
aws s3control create-access-grants-location \
 --account-id 111122223333 \
 --location-scope s3://DOC-BUCKET-EXAMPLE/ \
 --iam-role-arn arn:aws:iam::123456789012:role/accessGrantsTestRole
```

Antwort:

```
{"CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "635f1139-1af2-4e43-8131-a4de006eb456",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2: 111122223333:access-grants/
default/location/635f1139-1af2-4e43-8131-a4de006eb888",
    "LocationScope": "s3://DOC-BUCKET-EXAMPLE/",
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
}
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für die Verwaltung einer S3-Access-Grants-Instance finden Sie in den folgenden Abschnitten in der Amazon-Simple-Storage-Service-API-Referenz:

- CreateAccessGrantsLocation

- DeleteAccessGrantsLocation
- GetAccessGrantsLocation
- ListAccessGrantsLocations
- UpdateAccessGrantsLocation

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele für die Registrierung von Speicherorten unter Verwendung der AWS SDKs.

Um die folgenden Beispiele zu verwenden, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen.

Java

Sie können in Ihrer S3-Access-Grants-Instance den Standardspeicherort (`s3://`) oder einen benutzerdefinierten Speicherort registrieren. Sie müssen zunächst eine IAM-Rolle mit Prinzipalzugriff auf den Speicherort erstellen und dann S3 Access Grants die Berechtigung gewähren, diese Rolle anzunehmen.

Um die folgenden Beispielbefehle zu verwenden, ersetzen Sie *user input placeholders* durch eigene Daten.

Example Registrieren eines Standardspeicherorts

Anfrage:

```
public void createAccessGrantsLocation() {
CreateAccessGrantsLocationRequest createRequest =
 CreateAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.locationScope("s3://")
.iamRoleArn("arn:aws:iam::123456789012:role/accessGrantsTestRole")
.build();
CreateAccessGrantsLocationResponse createResponse =
 s3Control.createAccessGrantsLocation(createRequest);
LOGGER.info("CreateAccessGrantsLocationResponse: " + createResponse);
}
```

Antwort:

```
CreateAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:11.027Z,
AccessGrantsLocationId=default,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/default,
LocationScope=s3://,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
```

Example Registrieren eines benutzerdefinierten Speicherorts

Anfrage:

```
public void createAccessGrantsLocation() {
CreateAccessGrantsLocationRequest createRequest =
 CreateAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.locationScope("s3://DOC-BUCKET-EXAMPLE/")
.iamRoleArn("arn:aws:iam::111122223333:role/accessGrantsTestRole")
.build();
CreateAccessGrantsLocationResponse createResponse =
 s3Control.createAccessGrantsLocation(createRequest);
LOGGER.info("CreateAccessGrantsLocationResponse: " + createResponse);
}
```

Antwort:

```
CreateAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=18cfe6fb-eb5a-4ac5-aba9-8d79f04c2012,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/18cfe6fb-eb5a-4ac5-aba9-8d79f04c2666,
LocationScope= s3://test-bucket-access-grants-user123/,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
```

Themen

- Anzeigen der Details eines registrierten Speicherorts
- Aktualisieren eines registrierten Speicherorts
- Löschen eines registrierten Speicherorts

Anzeigen der Details eines registrierten Speicherorts

Sie können die Details zu einem Speicherort abrufen, der in Ihrer S3-Access-Grants-Instance registriert ist, indem Sie die Amazon S3-Konsole, die AWS Command Line Interface (AWS CLI), die Amazon S3-REST-API und die - AWS SDKs verwenden.

Verwenden der S3-Konsole

So zeigen Sie die die in Ihrer S3-Access-Grants-Instance gespeicherten Speicherorte an

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.
4. Wählen Sie Details anzeigen für die Instance aus.
5. Wählen Sie auf der Detailseite für die Instance die Registerkarte Speicherorte aus.
6. Suchen Sie den registrierten Speicherort, den Sie anzeigen möchten. Verwenden Sie das Suchfeld, um die Liste der registrierten Speicherorte zu filtern.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen.

Example – Abrufen der Details eines registrierten Speicherorts

```
aws s3control get-access-grants-location \
--account-id 111122223333 \
--access-grants-location-id default
```

Antwort:

```
{
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
```

```
    "AccessGrantsLocationId": "default",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/default",
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
}
```

Example – Auflisten aller in einer S3-Access-Grants-Instance registrierten Speicherorte

Um die Ergebnisse auf ein S3-Präfix oder einen S3-Bucket einzuschränken, können Sie optional den Parameter `--location-scope s3://bucket-and-or-prefix` verwenden.

```
aws s3control list-access-grants-locations \
--account-id 111122223333 \
--region us-east-2
```

Antwort:

```
{"AccessGrantsLocationsList": [
  {
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "default",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/default",
    "LocationScope": "s3://"
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
     },
  {
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "635f1139-1af2-4e43-8131-a4de006eb456",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/635f1139-1af2-4e43-8131-a4de006eb888",
    "LocationScope": "s3://DOC-EXAMPLE-BUCKET/prefixA*",
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
     }
   ]
  }
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für das Abrufen der Details eines registrierten Speicherorts oder für das Auflisten aller Speicherorte, die bei einer S3-Access-Grants-

Instance registriert sind, finden Sie in den folgenden Abschnitten der Amazon-Simple-Storage-Service-API-Referenz:

- GetAccessGrantsLocation
- ListAccessGrantsLocations

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele dafür, wie Sie über die AWS SDKs die Details eines registrierten Speicherorts abrufen oder alle in einer S3-Access-Grants-Instance registrierten Speicherorte auflisten können.

Um die folgenden Beispiele zu verwenden, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen.

Java

Example – Abrufen der Details eines registrierten Speicherorts

```
public void getAccessGrantsLocation() {
GetAccessGrantsLocationRequest getAccessGrantsLocationRequest =
 GetAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.accessGrantsLocationId("default")
.build();
GetAccessGrantsLocationResponse getAccessGrantsLocationResponse =
 s3Control.getAccessGrantsLocation(getAccessGrantsLocationRequest);
LOGGER.info("GetAccessGrantsLocationResponse: " + getAccessGrantsLocationResponse);
}
```

Antwort:

```
GetAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=default,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/default,
LocationScope= s3://,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
```

Example – Auflisten aller in einer S3-Access-Grants-Instance registrierten Speicherorte

Um die Ergebnisse auf ein S3-Präfix oder einen S3-Bucket einzuschränken, können Sie im Parameter `LocationScope` optional eine S3-URI übergeben, z. B. `s3://bucket-and-or-prefix`.

```
public void listAccessGrantsLocations() {

ListAccessGrantsLocationsRequest listRequest =
 ListAccessGrantsLocationsRequest.builder()
.accountId("111122223333")
.build();

ListAccessGrantsLocationsResponse listResponse =
 s3Control.listAccessGrantsLocations(listRequest);
LOGGER.info("ListAccessGrantsLocationsResponse: " + listResponse);
}
```

Antwort:

```
ListAccessGrantsLocationsResponse(
AccessGrantsLocationsList=[
ListAccessGrantsLocationsEntry(
CreatedAt=2023-06-07T04:35:11.027Z,
AccessGrantsLocationId=default,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/default,
LocationScope=s3://,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
),
ListAccessGrantsLocationsEntry(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=635f1139-1af2-4e43-8131-a4de006eb456,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/635f1139-1af2-4e43-8131-a4de006eb888,
LocationScope=s3://DOC-EXAMPLE-BUCKET/prefixA*,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
]
)
```

Aktualisieren eines registrierten Speicherorts

Sie können die AWS Identity and Access Management (IAM)-Rolle eines Speicherorts aktualisieren, der in Ihrer Amazon S3-Access-Grants-Instance registriert ist. Sie müssen für jede neue IAM-Rolle, die Sie für die Registrierung eines Speicherorts in S3 Access Grants verwenden, dem S3-Access-Grants-Service-Prinzipal (`access-grants.s3.amazonaws.com`) Zugriff auf diese Rolle gewähren. Hierzu fügen Sie einen Eintrag für die neue IAM-Rolle in derselben JSON-Vertrauensrichtliniendatei hinzu, die Sie bei der Registrierung des Speicherorts verwendet haben.

Sie können einen Speicherort in Ihrer S3-Access-Grants-Instance mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und der AWS SDKs aktualisieren.

Verwenden der S3-Konsole

So aktualisieren Sie die IAM-Rolle eines Speicherorts, der bei Ihrer S3-Access-Grants-Instance registriert ist

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.
4. Wählen Sie Details anzeigen für die Instance aus.
5. Wählen Sie auf der Detailseite für die Instance die Registerkarte Speicherorte aus.
6. Suchen Sie den Speicherort aus, den Sie aktualisieren möchten. Verwenden Sie das Suchfeld, um die Liste der Speicherorte zu filtern.
7. Wählen Sie die Optionsschaltfläche neben dem registrierten Speicherort aus, den Sie aktualisieren möchten.
8. Aktualisieren Sie die IAM-Rolle und wählen Sie dann Änderungen speichern aus.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen.

Example – Aktualisieren der IAM-Rolle eines registrierten Speicherorts

```
aws s3control update-access-grants-location \
--account-id 111122223333 \
--access-grants-location-id 635f1139-1af2-4e43-8131-a4de006eb999 \
--iam-role-arn arn:aws:iam::777788889999:role/accessGrantsTestRole
```

Antwort:

```
{
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "635f1139-1af2-4e43-8131-a4de006eb999",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:777788889999:access-grants/
default/location/635f1139-1af2-4e43-8131-a4de006eb888",
    "LocationScope": "s3://DOC-EXAMPLE-BUCKET/prefixB*",
    "IAMRoleArn": "arn:aws:iam::777788889999:role/accessGrantsTestRole"
}
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für das Aktualisieren eines Speicherorts in einer S3-Access-Grants-Instance finden Sie unter UpdateAccessGrantsLocation in der Amazon-Simple-Storage-Service-API-Referenz.

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele für die Aktualisierung der IAM-Rolle eines registrierten Speicherorts mithilfe der - AWS SDKs.

Wenn Sie das folgende Beispiel verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen.

Java

Example – Aktualisieren der IAM-Rolle eines registrierten Speicherorts

```
public void updateAccessGrantsLocation() {
UpdateAccessGrantsLocationRequest updateRequest =
 UpdateAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.accessGrantsLocationId("635f1139-1af2-4e43-8131-a4de006eb999")
.iamRoleArn("arn:aws:iam::777788889999:role/accessGrantsTestRole")
```

```
.build();
UpdateAccessGrantsLocationResponse updateResponse =
 s3Control.updateAccessGrantsLocation(updateRequest);
LOGGER.info("UpdateAccessGrantsLocationResponse: " + updateResponse);
}
```

Antwort:

```
UpdateAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=635f1139-1af2-4e43-8131-a4de006eb999,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:777788889999:access-grants/default/
location/635f1139-1af2-4e43-8131-a4de006eb888,
LocationScope=s3://DOC-EXAMPLE-BUCKET/prefixB*,
IAMRoleArn=arn:aws:iam::777788889999:role/accessGrantsTestRole
)
```

Löschen eines registrierten Speicherorts

Sie können eine Speicherortregistrierung aus einer Amazon-S3-Access-Grants-Instance löschen. Durch das Löschen des Speicherorts wird dessen Registrierung in der S3-Access-Grants-Instance aufgehoben.

Bevor Sie eine Speicherortregistrierung aus einer S3-Access-Grants-Instance entfernen können, müssen Sie alle mit diesem Speicherort verknüpften Gewährungen löschen. Informationen zum Löschen von Gewährungen finden Sie unter Löschen einer Gewährung.

Sie können einen Speicherort in Ihrer S3-Access-Grants-Instance mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und der AWS SDKs löschen.

Verwenden der S3-Konsole

So löschen Sie eine Speicherortregistrierung aus Ihrer Amazon-S3-Access-Grants-Instance

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.

4. Wählen Sie Details anzeigen für die Instance aus.
5. Wählen Sie auf der Detailseite für die Instance die Registerkarte Speicherorte aus.
6. Suchen Sie den Speicherort aus, den Sie aktualisieren möchten. Verwenden Sie das Suchfeld, um die Liste der Speicherorte zu filtern.
7. Wählen Sie die Optionsschaltfläche neben dem registrierten Speicherort aus, den Sie löschen möchten.
8. Wählen Sie Deregister.
9. Anschließend wird ein Dialogfeld angezeigt, in dem Sie gewarnt werden, dass diese Aktion nicht rückgängig gemacht werden kann. Um den Speicherort zu löschen, wählen Sie Registrierung aufheben aus.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

Example – Löschen einer Speicherortregistrierung

```
aws s3control delete-access-grants-location \
--account-id 111122223333 \
--access-grants-location-id  a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
 // No response body
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für das Löschen eines Speicherorts aus einer S3-Access-Grants-Instance finden Sie unter DeleteAccessGrantsLocation in der Amazon-Simple-Storage-Service-API-Referenz.

Verwenden der AWS SDKs

Dieser Abschnitt enthält ein Beispiel für das Löschen eines Speicherorts unter Verwendung der AWS SDKs.

Wenn Sie das folgende Beispiel verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen.

Java

Example – Löschen einer Speicherortregistrierung

```
public void deleteAccessGrantsLocation() {
DeleteAccessGrantsLocationRequest deleteRequest =
 DeleteAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.accessGrantsLocationId("a1b2c3d4-5678-90ab-cdef-EXAMPLE11111")
.build();
DeleteAccessGrantsLocationResponse deleteResponse =
 s3Control.deleteAccessGrantsLocation(deleteRequest);
LOGGER.info("DeleteAccessGrantsLocationResponse: " + deleteResponse);
}
```

Antwort:

```
DeleteAccessGrantsLocationResponse()
```

Erstellen von Gewährungen

Sie müssen in Ihrer S3-Access-Grants-Instance mindestens einen Speicherort registrieren, bevor Sie eine Zugriffsgewährung erstellen können. Eine Zugriffsgewährung gewährt dem Empfänger die Berechtigung, auf einen registrierten Speicherort zuzugreifen.

Der Empfänger kann ein AWS Identity and Access Management (IAM)-Benutzer oder eine IAM-Rolle oder ein Verzeichnisbenutzer oder eine Verzeichnisgruppe sein. Ein Verzeichnisbenutzer ist ein Benutzer aus Ihrem Unternehmensverzeichnis oder aus einer externen Identitätsquelle, den Sie zu der AWS IAM Identity Center -Instance hinzugefügt haben, die mit Ihrer S3-Access-Grants-Instance verknüpft ist. Um eine Gewährung für einen bestimmten Benutzer oder eine bestimmte Gruppe über IAM Identity Center zu erstellen, suchen Sie die GUID, mit der dieser Benutzer in IAM Identity Center identifiziert wird, zum Beispiel `a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`.

Sie können Zugriff auf einen Bucket, ein Präfix oder ein Objekt gewähren. Ein Präfix in Amazon S3 ist eine Zeichenfolge am Anfang eines Objektschlüsselnamens, mit der die Objekte in einem Bucket organisiert werden. Dabei kann es sich um eine beliebige Zeichenfolge handeln, z. B. um Objektschlüsselnamen in einem Bucket, die mit dem Präfix `engineering/` beginnen.

Unterpräfix

Wenn Sie Zugriff auf einen registrierten Speicherort gewähren, können Sie im Feld `Subprefix` den Umfang auf ein bestimmtes Präfix oder ein bestimmtes Objekt in einem Bucket eingrenzen.

Sie können keine Zugriffsgewährung für den Standardspeicherort in `s3://` erstellen, da der Empfänger hierdurch Zugriff auf alle Buckets in einer Region erhält. Wenn Sie den Standardspeicherort in `s3://` als Speicherort für die Gewährung auswählen, müssen Sie im Feld `Subprefix` den Umfang der Gewährung durch die Angabe eines der folgenden Elemente eingrenzen:

- Bucket – `s3://`*`bucket`*`/*`
- Präfix innerhalb eines Buckets – `s3://`*`bucket`*`/`*`prefix`*`*`
- Präfix innerhalb eines Präfixes – `s3://`*`bucket`*`/`*`prefixA`*`/`*`prefixB`*`*`
- Objekt – `s3://`*`bucket`*`/`*`object-key-name`*

Wenn Sie eine Zugriffsberechtigung erstellen, deren registrierter Speicherort ein Bucket ist, können Sie im Feld `Subprefix` eines der folgenden Elemente übergeben:

- Präfix innerhalb des Buckets – *`prefix`*`*`
- Präfix innerhalb eines Präfixes – *`prefixA/prefixB`*`*`
- Objekt – `/`*`object-key-name`*

Der in der Amazon S3-Konsole oder der in der API- oder AWS Command Line Interface (AWS CLI)-Antwort `GrantScope` zurückgegebene Erteilungsbereich ist das Ergebnis der Verkettung des Standortpfads mit der `Subprefix`. Dieser verkettete Pfad muss mit dem S3-Bucket, Präfix oder Objekt übereinstimmen, dem Sie Zugriff gewähren möchten.

Wenn Sie eine Zugriffsgewährung erstellen, die Zugriff auf nur ein Objekt gewährt, müssen Sie im API-Aufruf oder CLI-Befehl angeben, dass `s3PrefixType` `Object` ist.

> **Note**
>
> Sie können keine Gewährung für einen Bucket erstellen, der noch nicht vorhanden ist. Sie können jedoch eine Gewährung für ein Präfix erstellen, das noch nicht vorhanden ist.

Sie können eine Zugriffsgewährung mithilfe der Amazon S3-Konsole AWS CLI, der Amazon S3-REST-API und AWS SDKs erstellen.

Verwenden der S3-Konsole

So erstellen Sie eine Zugriffsgewährung

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.

   Wenn Sie die S3-Access-Grants-Instance zum ersten Mal verwenden, müssen Sie Schritt 2 – Registrieren eines Speicherorts abgeschlossen haben und zu Schritt 3 des Assistenten Access-Grants-Instance einrichten gewechselt sein. Wenn Sie bereits über eine S3-Access-Grants-Instance verfügen, wählen Sie Details anzeigen und dann auf der Registerkarte Gewährungen die Option Gewährung erstellen aus.

   a. Wählen Sie im Abschnitt Gewährungsumfang einen registrierten Standort aus oder geben Sie diesen ein.

      Wenn Sie den `s3://`-Standardspeicherort ausgewählt haben, können Sie im Feld Unterpräfix den Umfang der Zugriffsgewährung einschränken. Weitere Informationen finden Sie unter Unterpräfix. Wenn Sie lediglich einem Objekt Zugriff gewähren, wählen Sie Gewährungsumfang ist ein Objekt aus.

   b. Wählen Sie unter Berechtigungen und Zugriff die Stufe der Berechtigung aus, Lesen, Schreiben oder beides.

      Wählen Sie dann den Empfängertyp aus. Wenn Sie Ihr Unternehmensverzeichnis zu IAM Identity Center hinzugefügt und diese IAM-Identity-Center-Instance mit Ihrer S3-Access-Grants-Instance verknüpft haben, können Sie Verzeichnisidentität aus IAM Identity Center auswählen. Wenn Sie diese Option auswählen, rufen Sie die ID des Benutzers oder der Gruppe aus IAM Identity Center ab und geben diese in diesen Abschnitt ein.

      Wenn der Empfängertyp ein IAM-Benutzer oder eine IAM-Rolle ist, wählen Sie IAM-Prinzipal aus. Wählen Sie in IAM-Prinzipaltyp die Option Benutzer oder Rolle aus. Wählen Sie dann in IAM-Prinzipalbenutzer entweder einen Eintrag aus der Liste aus oder geben Sie die ID der Identität ein.

c. Um die S3-Access-Grants-Gewährung zu erstellen, wählen Sie Weiter oder Gewährung erstellen aus.

4. Wenn Weiter oder Gewährung erstellen deaktiviert ist:

   Gewährung kann nicht erstellt werden

   - Möglicherweise müssen Sie in Ihrer S3-Access-Grants-Instance zuerst einen Speicherort registrieren.
   - Möglicherweise besitzen Sie die Berechtigung `s3:CreateAccessGrant` nicht, die für die Erstellung Zugriffsgewährung erforderlich ist. Nehmen Sie Kontakt mit Ihrem Kontoadministrator auf.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Die folgenden Beispiele zeigen, wie Sie eine Anforderung für eine Zugriffsgewährung für einen IAM-Prinzipal bzw. für einen Benutzer oder eine Gruppe im Unternehmensverzeichnis erstellen.

Um die folgenden Beispielbefehle zu verwenden, ersetzen Sie *`user input placeholders`* durch eigene Daten.

> **Note**
>
> Wenn Sie eine Zugriffsgewährung erstellen, die Zugriff auf ein einzelnes Objekt gewährt, geben Sie den erforderlichen Parameter `--s3-prefix-type Object` an.

Example Erstellen einer Anforderung für eine Zugriffsgewährung für einen IAM-Prinzipal

```
aws s3control create-access-grant \
--account-id 111122223333 \
--access-grants-location-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \
--access-grants-location-configuration S3SubPrefix=prefixB* \
--permission READ \
--grantee GranteeType=IAM,GranteeIdentifier=arn:aws:iam::123456789012:user/data-consumer-3
```

Example Erstellen einer Zugriffsgewährungsantwort

```
{"CreatedAt": "2023-05-31T18:41:34.663000+00:00",
    "AccessGrantId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Grantee": {
        "GranteeType": "IAM",
        "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-3"
    },
    "AccessGrantsLocationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "AccessGrantsLocationConfiguration": {
        "S3SubPrefix": "prefixB*"
    },
    "GrantScope": "s3://DOC-BUCKET-EXAMPLE/prefix*",
    "Permission": "READ"
}
```

Erstellen einer Anforderung für eine Zugriffsgewährung für einen Verzeichnisbenutzer oder eine Verzeichnisgruppe

Zur Erstellung einer Anforderung für eine Zugriffsgewährung für einen Verzeichnisbenutzer oder eine Verzeichnisgruppe müssen Sie zunächst die GUID für den Verzeichnisbenutzer oder die Verzeichnisgruppe abrufen, indem Sie einen der folgenden Befehle ausführen.

Example Abrufen einer GUID für einen Verzeichnisbenutzer oder eine Verzeichnisgruppe

Sie finden die GUID eines IAM-Identity-Center-Benutzers über die IAM-Identity-Center-Konsole oder mithilfe der AWS CLI oder AWS SDKs . Der folgende Befehl listet die Benutzer in der angegebenen IAM-Identity-Center-Instance mit Namen und IDs auf.

```
aws identitystore list-users --identity-store-id d-1a2b3c4d1234
```

Dieser Befehl listet die Gruppen in der angegebenen IAM-Identity-Center-Instance auf.

```
aws identitystore list-groups --identity-store-id d-1a2b3c4d1234
```

Example Erstellen einer Zugriffsgewährung für einen Verzeichnisbenutzer oder eine Verzeichnisgruppe

Dieser Befehl ist der Erstellung einer Gewährung für IAM-Benutzer oder -Rollen ähnlich. Der Empfängertyp ist jedoch `DIRECTORY_USER` oder `DIRECTORY_GROUP` und die Empfänger-ID ist die GUID für den Verzeichnisbenutzer oder die Verzeichnisgruppe.

```
aws s3control create-access-grant \
--account-id 123456789012 \
--access-grants-location-id default \
--access-grants-location-configuration S3SubPrefix="DOC-EXAMPLE-BUCKET/rafael/*" \
--permission READWRITE \
--grantee GranteeType=DIRECTORY_USER,GranteeIdentifier=83d43802-00b1-7054-db02-
f1d683aacba5 \
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für die Verwaltung von Zugriffsgewährungen finden Sie in den folgenden Abschnitten in der Amazon-Simple-Storage-Service-API-Referenz:

- CreateAccessGrant
- DeleteAccessGrant
- GetAccessGrant
- ListAccessGrants

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele dafür, wie Sie mit den AWS SDKs eine Zugriffsgewährung erstellen.

Java

> Wenn Sie das folgende Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch eigene Daten.

> **Note**
>
> Wenn Sie eine Zugriffsgewährung erstellen, die Zugriff auf ein einzelnes Objekt gewährt, geben Sie den erforderlichen Parameter `.s3PrefixType(S3PrefixType.Object)` an.

Example Erstellen einer Anforderung für eine Zugriffsgewährung

```
public void createAccessGrant() {
CreateAccessGrantRequest createRequest = CreateAccessGrantRequest.builder()
.accountId("111122223333")
.accessGrantsLocationId("a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa")
.permission("READ")
.accessGrantsLocationConfiguration(AccessGrantsLocationConfiguration.builder().s3SubPrefix("
.grantee(Grantee.builder().granteeType("IAM").granteeIdentifier("arn:aws:iam::111122223333:u
data-consumer-3").build())
.build();
CreateAccessGrantResponse createResponse =
 s3Control.createAccessGrant(createRequest);
LOGGER.info("CreateAccessGrantResponse: " + createResponse);
}
```

Example Erstellen einer Zugriffsgewährungsantwort

```
CreateAccessGrantResponse(
CreatedAt=2023-06-07T05:20:26.330Z,
AccessGrantId=a1b2c3d4-5678-90ab-cdef-EXAMPLE33333,
AccessGrantArn=arn:aws:s3:us-east-2:444455556666:access-grants/default/grant/
a1b2c3d4-5678-90ab-cdef-EXAMPLE33333,
Grantee=Grantee(
GranteeType=IAM,
GranteeIdentifier=arn:aws:iam::111122223333:user/data-consumer-3
),
AccessGrantsLocationId=a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa,
AccessGrantsLocationConfiguration=AccessGrantsLocationConfiguration(
S3SubPrefix=prefixB*
),
GrantScope=s3://DOC-BUCKET-EXAMPLE/prefixB,
Permission=READ
)
```

Themen

Anzeigen einer Gewährung

Sie können die Details einer Zugriffsgewährung in Ihrer Amazon S3-Access-Grants-Instance mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und der AWS SDKs anzeigen.

Verwenden der S3-Konsole

So zeigen Sie die Details einer Zugriffsgewährung an

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.
4. Wählen Sie Details anzeigen für die Instance aus.
5. Wählen Sie auf der Detailseite die Registerkarte Gewährungen aus.
6. Suchen Sie im Bereich Gewährungen nach der Zugriffsgewährung, die Sie anzeigen möchten. Sie können die Liste der Gewährungen über das Suchfeld filtern.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Um die folgenden Beispielbefehle zu verwenden, ersetzen Sie *`user input placeholders`* durch eigene Daten.

Example – Abrufen der Details einer Zugriffsgewährung

```
aws s3control get-access-grant \
--account-id 111122223333 \
```

```
--access-grant-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

Antwort:

```
{
    "CreatedAt": "2023-05-31T18:41:34.663000+00:00",
    "AccessGrantId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Grantee": {
        "GranteeType": "IAM",
        "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-3"
    },
    "Permission": "READ",
    "AccessGrantsLocationId": "12a6710f-5af8-41f5-b035-0bc795bf1a2b",
    "AccessGrantsLocationConfiguration": {
        "S3SubPrefix": "prefixB*"
    },
    "GrantScope": "s3://DOC-EXAMPLE-BUCKET/"
}
```

Example – Auflisten aller Zugriffsgewährungen in einer S3-Access-Grants-Instance

Sie können optional die folgenden Parameter verwenden, um die Ergebnisse auf ein S3-Präfix oder eine AWS Identity and Access Management (IAM)-Identität zu beschränken:

- Unterpräfix – `--grant-scope s3://bucket-name/prefix*`
- IAM-Identität – `--grantee-type IAM` und `--grantee-identifier arn:aws:iam::123456789000:role/accessGrantsConsumerRole`

```
aws s3control list-access-grants \
--account-id 111122223333
```

Antwort:

```
{
    "AccessGrantsList": [{"CreatedAt": "2023-06-14T17:54:46.542000+00:00",
            "AccessGrantId": "dd8dd089-b224-4d82-95f6-975b4185bbaa",
            "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/dd8dd089-b224-4d82-95f6-975b4185bbaa",
```

```
            "Grantee": {
                "GranteeType": "IAM",
                "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-3"
            },
            "Permission": "READ",
            "AccessGrantsLocationId": "23514a34-ea2e-4ddf-b425-d0d4bfcarda1",
            "GrantScope": "s3://DOC-EXAMPLE-BUCKET/prefixA*"
        },
        {"CreatedAt": "2023-06-24T17:54:46.542000+00:00",
            "AccessGrantId": "ee8ee089-b224-4d72-85f6-975b4185a1b2",
            "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/ee8ee089-b224-4d72-85f6-975b4185a1b2",
            "Grantee": {
                "GranteeType": "IAM",
                "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-9"
            },
            "Permission": "READ",
            "AccessGrantsLocationId": "12414a34-ea2e-4ddf-b425-d0d4bfcacao0",
            "GrantScope": "s3://DOC-EXAMPLE-BUCKET/prefixB*"
        },

    ]
}
```

Verwenden der REST-API

Sie können Amazon-S3-API-Operationen verwenden, um die Details einer Zugriffsgewährung anzuzeigen und alle Zugriffsgewährungen in einer S3-Access-Grants-Instance aufzulisten. Informationen zur REST-API-Unterstützung für die Verwaltung von Zugriffsgewährungen finden Sie in den folgenden Abschnitten in der Amazon-Simple-Storage-Service-API-Referenz:

- GetAccessGrant
- ListAccessGrants

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele dafür, wie Sie die Details einer Zugriffsgewährung mithilfe der - AWS SDKs abrufen.

Um die folgenden Beispiele zu verwenden, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen.

Java

Example – Abrufen der Details einer Zugriffsgewährung

```
public void getAccessGrant() {
GetAccessGrantRequest getRequest = GetAccessGrantRequest.builder()
.accountId("111122223333")
.accessGrantId("a1b2c3d4-5678-90ab-cdef-EXAMPLE22222")
.build();
GetAccessGrantResponse getResponse = s3Control.getAccessGrant(getRequest);
LOGGER.info("GetAccessGrantResponse: " + getResponse);
}
```

Antwort:

```
GetAccessGrantResponse(
CreatedAt=2023-06-07T05:20:26.330Z,
AccessGrantId=a1b2c3d4-5678-90ab-cdef-EXAMPLE22222,
AccessGrantArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant-fd3a5086-42f7-4b34-9fad-472e2942c70e,
Grantee=Grantee(
GranteeType=IAM,
GranteeIdentifier=arn:aws:iam::111122223333:user/data-consumer-3
),
Permission=READ,
AccessGrantsLocationId=12a6710f-5af8-41f5-b035-0bc795bf1a2b,
AccessGrantsLocationConfiguration=AccessGrantsLocationConfiguration(
S3SubPrefix=prefixB*
),
GrantScope=s3://DOC-EXAMPLE-BUCKET/
)
```

Example – Auflisten aller Zugriffsgewährungen in einer S3-Access-Grants-Instance

Sie können optional die folgenden Parameter verwenden, um die Ergebnisse auf ein S3-Präfix oder eine IAM-Identität einzuschränken:

- Umfang – `GrantScope=s3://bucket-name/prefix*`
- Gewährungsempfänger – `GranteeType=IAM` und `GranteeIdentifier=arn:aws:iam::111122223333:role/accessGrantsConsumerRole`

```
public void listAccessGrants() {
ListAccessGrantsRequest listRequest = ListAccessGrantsRequest.builder()
.accountId("111122223333")
.build();
ListAccessGrantsResponse listResponse = s3Control.listAccessGrants(listRequest);
LOGGER.info("ListAccessGrantsResponse: " + listResponse);
}
```

Antwort:

```
ListAccessGrantsResponse(
AccessGrantsList=[
ListAccessGrantEntry(
CreatedAt=2023-06-14T17:54:46.540z,
AccessGrantId=dd8dd089-b224-4d82-95f6-975b4185bbaa,
AccessGrantArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/dd8dd089-b224-4d82-95f6-975b4185bbaa,
Grantee=Grantee(
GranteeType=IAM, GranteeIdentifier= arn:aws:iam::111122223333:user/data-consumer-3
),
Permission=READ,
AccessGrantsLocationId=23514a34-ea2e-4ddf-b425-d0d4bfcarda1,
GrantScope=s3://DOC-EXAMPLE-BUCKET/prefixA
),
ListAccessGrantEntry(
CreatedAt=2023-06-24T17:54:46.540Z,
AccessGrantId=ee8ee089-b224-4d72-85f6-975b4185a1b2,
AccessGrantArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/ee8ee089-b224-4d72-85f6-975b4185a1b2,
Grantee=Grantee(
GranteeType=IAM, GranteeIdentifier= arn:aws:iam::111122223333:user/data-consumer-9
),
Permission=READ,
AccessGrantsLocationId=12414a34-ea2e-4ddf-b425-d0d4bfcacao0,
GrantScope=s3://DOC-EXAMPLE-BUCKET/prefixB*
)
]
)
```

Löschen einer Gewährung

Sie können Zugriffsgewährungen aus Ihrer Amazon-S3-Access-Grants-Instance löschen. Sie können die Löschung einer Zugriffsgewährung nicht rückgängig machen. Nach dem Löschen einer Zugriffsberechtigung hat der Empfänger nicht länger Zugriff auf Ihre Amazon-S3-Daten.

Sie können eine Zugriffsgewährung mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und der AWS SDKs löschen.

Verwenden der S3-Konsole

So löschen Sie eine Zugriffsgewährung

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Zugriffsgewährungen aus.
3. Wählen Sie auf der Seite S3 Access Grants die Region mit der S3-Access-Grants-Instance aus, mit der Sie arbeiten möchten.
4. Wählen Sie Details anzeigen für die Instance aus.
5. Wählen Sie auf der Detailseite die Registerkarte Gewährungen aus.
6. Suchen Sie die Gewährung, die Sie löschen möchten. Wenn Sie die Gewährung gefunden haben, wählen Sie das Optionsfeld neben der Gewährung aus.
7. Wählen Sie Löschen aus. Anschließend wird ein Dialogfeld angezeigt, in dem Sie gewarnt werden, dass diese Aktion nicht rückgängig gemacht werden kann. Wählen Sie erneut Löschen aus, um die Gewährung zu löschen.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen.

Example – Löschen einer Zugriffsgewährung

```
aws s3control delete-access-grant \
--account-id 111122223333 \
```

```
--access-grant-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111

// No response body
```

Verwenden der REST-API

Informationen zur Amazon-S3-REST-API-Unterstützung für das Verwalten von Zugriffsgewährungen finden Sie unter DeleteAccessGrant in der Amazon-Simple-Storage-Service-API-Referenz.

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele für das Löschen einer Zugriffsgewährung mithilfe der - AWS SDKs. Wenn Sie das folgende Beispiel verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen.

Java

Example – Löschen einer Zugriffsgewährung

```
public void deleteAccessGrant() {
DeleteAccessGrantRequest deleteRequest = DeleteAccessGrantRequest.builder()
.accountId("111122223333")
.accessGrantId("a1b2c3d4-5678-90ab-cdef-EXAMPLE11111")
.build();
DeleteAccessGrantResponse deleteResponse =
 s3Control.deleteAccessGrant(deleteRequest);
LOGGER.info("DeleteAccessGrantResponse: " + deleteResponse);
}
```

Antwort:

```
DeleteAccessGrantResponse()
```

Anfordern des Zugriffs auf Amazon-S3-Daten über S3 Access Grants

Nachdem Sie Amazon S3 Access Grants verwendet haben, um eine Zugriffsgewährung zu erstellen, die AWS Identity and Access Management (IAM)-Prinzipalen, Ihren Unternehmensverzeichnisidentitäten oder autorisierten Anwendungen Zugriff auf Ihre S3-Daten gewährt, können Ihre Berechtigungsempfänger Anmeldeinformationen für den Zugriff auf diese Daten anfordern.

Wenn eine Anwendung oder die API`GetDataAccess`-Operation AWS-Service verwendet, um S3 Access Grants im Namen eines Berechtigungsempfängers um den Zugriff auf Ihre S3-Daten zu bitten, überprüft S3 Access Grants zunächst, ob Sie dieser Identität Zugriff auf die Daten gewährt haben. Anschließend verwendet S3 Access Grants die API-Operation AssumeRole, um die dem registrierten Datenspeicherort zugeordnete IAM-Rolle zu übernehmen. Anschließend ruft die S3-Access-Grants-Funktion ein Token für temporäre Anmeldeinformationstoken ab und gibt es an den Anforderer zurück. Dieses Token für temporäre Anmeldeinformation ist ein AWS Security Token Service (AWS STS)-Token.

Die `GetDataAccess`-Anforderung muss den Parameter `target` enthalten, um den Umfang der S3-Daten anzugeben, für den die temporären Anmeldeinformationen gelten. Dieser `target`-Umfang kann mit dem Umfang der Gewährung oder einem Teil dieses Umfangs identisch sein. Der `target`-Umfang muss jedoch innerhalb des Umfangs der Gewährung für den Anforderer liegen. In der Anforderung muss auch der Parameter `permission` angegeben werden, der die Berechtigungsstufe für die temporären Anmeldeinformationen angibt, `READ`, `WRITE` oder `READWRITE`.

Der Anforderer kann die Berechtigungsstufe des temporären Tokens in der Anforderung für Anmeldeinformationen angeben. Mit dem Parameter `privilege` kann der Anforderer den Umfang des Zugriffs für die temporären Anmeldeinformationen innerhalb der Grenzen des Gewährungsumfangs reduzieren oder erhöhen. Der Standardwert des Parameters `privilege` ist `Default`. Das bedeutet, dass der Zielumfang der zurückgegebenen Anmeldeinformationen der ursprüngliche Umfang der Gewährung ist. Der andere mögliche Wert für `privilege` ist `Minimal`. Wenn der `target`-Umfang im Vergleich zum ursprünglichen Umfang der Gewährung reduziert wird, wird der Umfang der temporären Anmeldeinformationen an den `target`-Umfang angepasst, solange der `target`-Umfang innerhalb des Gewährungsumfangs liegt.

Die folgende Tabelle zeigt die Auswirkungen des Parameters `privilege` auf zwei Gewährungen. Eine Gewährung hat den Umfang `S3://`*`DOC-EXAMPLE-BUCKET1`/bob/*. Dieser Umfang umfasst das gesamte Präfix bob/ im Bucket *`DOC-EXAMPLE-BUCKET1`*. Die andere Gewährung hat den Umfang `S3://`*`DOC-EXAMPLE-BUCKET1`*`/bob/reports/*`. Dieser Umfang umfasst nur das Präfix `bob/reports/` im Bucket *`DOC-EXAMPLE-BUCKET1`*.

Umfang der Gewährung	Angeforderter Umfang	Recht	Zurückgegebener Umfang	Auswirkung
`S3://`*`DOC-EXAMP`*	*`DOC-EXAMP`*	`Default`	*`DOC-EXAMPLE-BUCKET1`* `/bob/*`	Der Anforderer hat Zugriff auf alle Objekte

Umfang der Gewährung	Angeforderter Umfang	Recht	Zurückgegebener Umfang	Auswirkung
LE-BUCKET 1 /bob/*	*LE-BUCKET 1* /bob/*			im Bucket *DOC-EXAMPLE-BUCKET1* , deren Schlüsselnamen mit dem Präfix *bob/* beginnen.
S3://*DOC-EXAMPLE-BUCKET 1* /bob/*	*DOC-EXAMPLE-BUCKET 1* /bob/	Minimal	*DOC-EXAMPLE-BUCKET 1* /bob/	Ohne das Platzhalterzeichen (*) nach dem Präfixnamen bob/ hat der Anforderer nur Zugriff auf das Objekt mit dem Namen bob/ im Bucket *DOC-EXAMPLE-BUCKET1* . Ein solches Objekt ist nicht üblich. Der Anforderer hat keinen Zugriff auf andere Objekte, auch nicht auf Objekte, deren Schlüsselnamen mit dem Präfix bob/ beginnen.
S3://*DOC-EXAMPLE-BUCKET 1* /bob/*	*DOC-EXAMPLE-BUCKET 1* /bob/images/*	Minimal	*DOC-EXAMPLE-BUCKET 1* /bob/images/*	Der Anforderer hat Zugriff auf alle Objekte im Bucket *DOC-EXAMPLE-BUCKET1* , deren Schlüsselnamen mit dem Präfix *bob/images/** beginnen.

Umfang der Gewährung	Angeforderter Umfang	Recht	Zurückgegebener Umfang	Auswirkung
S3://*DOC-EXAMPLE-BUCKET1* /bob/reports/*	*DOC-EXAMPLE-BUCKET1* /bob/reports/file.txt	Default	*DOC-EXAMPLE-BUCKET1* /bob/reports/*	Der Anforderer hat Zugriff auf alle Objekte im Bucket *DOC-EXAMPLE-BUCKET1* , deren Schlüsselnamen mit dem Präfix bob/reports beginnen. Dies entspricht dem Umfang der entsprechenden Gewährung.
S3://*DOC-EXAMPLE-BUCKET1* /bob/reports/*	*DOC-EXAMPLE-BUCKET1* /bob/reports/file.txt	Minimal	*DOC-EXAMPLE-BUCKET1* /bob/reports/file.txt	Der Anforderer hat nur Zugriff auf das Objekt mit dem Schlüsselnamen bob/reports/file.txt im Bucket *DOC-EXAMPLE-BUCKET1* . Der Anforderer hat keinen Zugriff auf andere Objekte.

Der Parameter `durationSeconds` legt die Gültigkeitsdauer der temporären Anmeldeinformationen in Sekunden fest. Der Standardwert ist `3600` Sekunden (1 Stunde). Der Anforderer (Empfänger) kann jedoch einen Bereich von `900` Sekunden (15 Minuten) bis `43200` Sekunden (12 Stunden) angeben. Wenn der Empfänger einen höheren Wert als diesen Höchstwert anfordert, schlägt die Anforderung fehl.

> **Note**
>
> Wenn der Speicherort ein Objekt ist, legen Sie in Ihrer Anforderung eines temporären Tokens den Wert des Parameters `targetType` auf `Object` fest. Dieser Parameter ist nur erforderlich, wenn es sich der Speicherort ein Objekt ist und die Berechtigungsstufe `Minimal`

ist. Wenn der Speicherort ein Bucket oder ein Präfix ist, müssen Sie diesen Parameter nicht angeben.

Weitere Informationen finden Sie unter GetDataAccess in der API-Referenz zu Amazon Simple Storage Service.

Sie können temporäre Anmeldeinformationen über AWS Command Line Interface (AWS CLI), die Amazon S3-REST-API und die AWS SDKs anfordern.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

Example Anfordern temporärer Anmeldeinformationen

Anfrage:

```
aws s3control get-data-access \
--account-id 111122223333 \
--target s3://DOC-EXAMPLE-BUCKET/prefixA* \
--permission READ \
--privilege Default \
--region us-east-2
```

Antwort:

```
{
"Credentials": {
 "AccessKeyId": "Example-key-id",
 "SecretAccessKey": "Example-access-key",
 "SessionToken": "Example-session-token",
"Expiration": "2023-06-14T18:56:45+00:00"},
    "MatchedGrantTarget": "s3://DOC-EXAMPLE-BUCKET/prefixA**"
}
```

Verwenden der REST-API

Informationen zur Amazon S3-REST-API-Unterstützung für das Anfordern temporärer Anmeldeinformationen von S3 Access Grants finden Sie unter GetDataAccess in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS SDKs

Dieser Abschnitt enthält ein Beispiel dafür, wie Berechtigungsempfänger mithilfe der SDKs temporäre Anmeldeinformationen von S3 Access AWS SDKs anfordern.

Java

Das folgende Codebeispiel gibt die temporären Anmeldeinformationen zurück, die der Gewährungsempfänger für den Zugriff auf Ihre S3-Daten verwendet. Wenn Sie dieses Codebeispiel verwenden möchten, ersetzen Sie *user input placeholders* durch eigene Daten.

Example Abrufen temporärer Anmeldeinformationen

Anfrage:

```
public void getDataAccess() {
GetDataAccessRequest getDataAccessRequest = GetDataAccessRequest.builder()
.accountId("111122223333")
.permission(Permission.READ)
.privilege(Privilege.MINIMAL)
.target("s3://DOC-EXAMPLE-BUCKET/prefixA*")
.build();
GetDataAccessResponse getDataAccessResponse =
 s3Control.getDataAccess(getDataAccessRequest);
LOGGER.info("GetDataAccessResponse: " + getDataAccessResponse);
}
```

Antwort:

```
GetDataAccessResponse(
Credentials=Credentials(
AccessKeyId="Example-access-key-id",
SecretAccessKey="Example-secret-access-key",
SessionToken="Example-session-token",
Expiration=2023-06-07T06:55:24Z
```

```
))
```

Zugriff auf S3-Daten über eine Zugriffsgewährung

Nachdem ein Gewährungsempfänger über seine Zugriffsgewährung temporäre Anmeldeinformationen erhalten hat, kann er mit diesen temporären Anmeldeinformationen Amazon-S3-API-Operationen für den Zugriff auf Ihre Daten aufrufen.

Empfänger können über die AWS Command Line Interface (AWS CLI), die - AWS SDKs und die Amazon-S3-REST-API auf S3-Daten zugreifen. Amazon S3

Verwenden der AWS CLI

Nachdem der Gewährungsempfänger seine temporären Anmeldeinformationen von S3 Access Grants erhalten hat, kann er mit diesen Anmeldeinformationen ein Profil einrichten, um die Daten abzurufen.

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Um die folgenden Beispielbefehle zu verwenden, ersetzen Sie *user input placeholders* durch eigene Daten.

Example – Einrichten eines Profils

```
aws configure set aws_access_key_id "$accessKey" --profile access-grants-consumer-access-profile
aws configure set aws_secret_access_key "$secretKey" --profile access-grants-consumer-access-profile
aws configure set aws_session_token "$sessionToken" --profile access-grants-consumer-access-profile
```

Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

Example – Abrufen der S3-Daten

Der Berechtigungsempfänger kann den get-object AWS CLI Befehl verwenden, um auf die Daten zuzugreifen. Der Berechtigungsempfänger kann auch put-object, lsund andere S3 AWS CLI -Befehle verwenden.

```
aws s3api get-object \
--bucket DOC-EXAMPLE-BUCKET1 \
--key myprefix \
--region us-east-2 \
--profile access-grants-consumer-access-profile
```

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele dafür, wie Gewährungsempfänger über die AWS SDKs auf Ihre S3-Daten zugreifen können.

Java

Beispiele für das Abrufen von S3-Daten mithilfe temporärer Anmeldeinformationen finden Sie unter Abrufen eines Objekts mithilfe der AWS SDKs und Amazon S3-Codebeispiele für die AWS SDK for Java 2.x.

Kontoübergreifender Zugriff mit S3 Access Grants

Die S3-Access-Grants-Instance selbst unterstützt ressourcenbasierte Richtlinien. Wenn also die richtigen ressourcenbasierten Richtlinien vorhanden sind, können Sie AWS Identity and Access Management (IAM)-Benutzern oder -Rollen von anderen AWS-Konten Zugriff auf Ihre S3-Access-Grants-Instance gewähren. Der kontoübergreifende Zugriff unterstützt nur folgende Berechtigungen:

- `s3:GetAccessGrantsInstanceForPrefix` – Ermöglicht Ihnen das Abrufen einer S3-Access-Grants-Instance, die ein bestimmtes Präfix enthält.
- `s3:ListAccessGrants`
- `s3:ListAccessLocations`
- `s3:GetDataAccess` – Ermöglicht Ihnen das Anfordern temporäre Anmeldeinformationen auf Grundlage des Zugriffs, der Ihnen über S3 Access Grants gewährt wurde. Mit diesen Anmeldeinformationen können Sie auf die S3-Daten zugreifen, für die Ihnen Zugriff gewährt wurde.

Sie können auswählen, welche dieser Berechtigungen in die Ressourcenrichtlinie aufgenommen werden sollen.

Diese Ressourcenrichtlinie für die S3-Access-Grants-Instance ist eine reguläre ressourcenbasierte Richtlinie und unterstützt alles, was auch die IAM-Richtliniensprache unterstützt. Sie können beispielsweise in derselben Richtlinie bestimmten IAM-Prinzipalen im Konto 111122223333 mit

der Bedingung `aws:PrincipalArn` Zugriff gewähren. Mit S3 Access Grants ist dies jedoch nicht erforderlich. Stattdessen würden Sie innerhalb Ihrer S3-Access-Grants-Instance Gewährungen für einzelne IAM-Prinzipale in diesem Konto erstellen. Da Sie jede Zugriffsgewährung einzeln über S3 Access Grants verwalten, können Sie Ihre Berechtigungen skalieren.

Sie können AWS Resource Access Manager (AWS RAM) verwenden, um Ihre `s3:AccessGrants`-Ressourcen mit anderen Konten oder innerhalb Ihrer Organisation zu teilen. Weitere Informationen finden Sie unter Arbeiten mit freigegebenen AWS Ressourcen. Wenn Sie nicht verwenden AWS RAM, können Sie die Ressourcenrichtlinie auch mithilfe der S3-Access-Grants-API-Operationen und der AWS Command Line Interface (AWS CLI) hinzufügen.

Sie können den kontoübergreifenden Zugriff auf eine S3-Access-Grants-Instance mithilfe der AWS Command Line Interface (AWS CLI), der Amazon S3-REST-API und AWS SDKs verwalten.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Example Hinzufügen oder Aktualisieren einer S3-Access-Grants-Ressourcenrichtlinie

Die folgende `resourcePolicy.json`-Datei ist ein Beispiel für eine S3-Access-Grants-Ressourcenrichtlinie, die dem Konto *123456789012* Zugriff auf die S3-Access-Grants-Instance im Konto *777788889999* gewährt. Wenn Sie diese Beispielrichtlinie verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

resourcePolicy.json

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
          "Principal": {
             "AWS": "123456789012"
          },
  "Action": [
        "s3:ListAccessGrants",
        "s3:ListAccessGrantsLocations",
        "s3:GetDataAccess"
        ],
```

```
   "Resource": "arn:aws:s3:us-east-2:777788889999:access-grants/default"
    }
  ]
}
```

Sie können den folgenden Beispielbefehl verwenden, um eine S3-Access-Grants-Ressourcenrichtlinie hinzuzufügen oder zu aktualisieren. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control put-access-grants-instance-resource-policy \
--account-id 777788889999 \
--policy file://resourcePolicy.json \
--profile access-grants-profile \
--region us-east-2

{
    "Policy": "{\n
   \"Version\": \"2012-10-17\",\n
   \"Statement\": [{\n
     \"Effect\": \"Allow\",\n
     \"Principal\": {\n
       \"AWS\": \"123456789012\"\n
     },\n
     \"Action\": [\n
       \"s3:ListAccessGrants\",\n
       \"s3:ListAccessGrantsLocations\",\n
       \"s3:GetDataAccess\"\n
     ],\n
     \"Resource\": \"arn:aws:s3:us-east-2:777788889999:access-grants/default\"\n
    }\n
   ]\n
   }\n",
    "CreatedAt": "2023-06-16T00:07:47.473000+00:00"
}
```

Example Abrufen einer S3-Access-Grants-Ressourcenrichtlinie

Sie können den folgenden Beispielbefehl verwenden, um eine S3-Access-Grants-Ressourcenrichtlinie abzurufen. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control get-access-grants-instance-resource-policy \
```

```
 --account-id 777788889999 \
 --profile access-grants-profile \
 --region us-east-2

{
    "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow
\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:root\"},\"Action\":
[\"s3:ListAccessGrants\",\"s3:ListAccessGrantsLocations\",\"s3:GetDataAccess\"],
\"Resource\":\"arn:aws:s3:us-east-2:777788889999:access-grants/default\"}]}",
    "CreatedAt": "2023-06-16T00:07:47.473000+00:00"
}
```

Example Löschen einer S3-Access-Grants-Ressourcenrichtlinie

Sie können den folgenden Beispielbefehl verwenden, um eine S3-Access-Grants-Ressourcenrichtlinie zu löschen. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control delete-access-grants-instance-resource-policy \
 --account-id 777788889999 \
 --profile access-grants-profile \
 --region us-east-2

// No response body
```

Verwenden der REST-API

Sie können die Amazon-S3-REST-API verwenden, um anderen Konten Zugriff auf Ihre S3-Access-Grants-Instance zu gewähren. Informationen zur REST-API-Unterstützung für den kontoübergreifenden Zugriff mit S3 Access Grants finden Sie in den folgenden Themen in der Amazon-Simple-Storage-Service-API-Referenz:

- PutAccessGrantsInstanceResourcePolicy
- GetAccessGrantsInstanceResourcePolicy
- DeleteAccessGrantsInstanceResourcePolicy

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele dafür, wie Sie mithilfe der -SDKs kontoübergreifenden Zugriff auf eine S3-Access- AWS SDKsInstance gewähren.

Java

Sie können eine Ressourcenrichtlinie hinzufügen, aktualisieren, abrufen oder löschen, um den kontoübergreifenden Zugriff für Ihre S3-Access-Grants-Instance zu verwalten.

Example Hinzufügen oder Aktualisieren einer S3-Access-Grants-Ressourcenrichtlinie

Die folgende `resourcePolicy.json`-Datei ist ein Beispiel für eine S3-Access-Grants-Ressourcenrichtlinie, die dem Konto *444455556666* Zugriff auf die S3-Access-Grants-Instance im Konto *111122223333* gewährt. Wenn Sie diese Beispielrichtlinie verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

resourcePolicy.json

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
          "Principal": {
             "AWS": "444455556666"
          },
  "Action": [
        "s3:ListAccessGrants",
        "s3:ListAccessGrantsLocations",
        "s3:GetDataAccess"
        ],
   "Resource": "arn:aws:s3:us-east-2:111122223333:access-grants/default"
    }
  ]
}
```

Sie können das folgende Codebeispiel verwenden, um eine S3-Access-Grants-Ressourcenrichtlinie hinzuzufügen oder zu aktualisieren.

```
public void putAccessGrantsInstanceResourcePolicy() {
PutAccessGrantsInstanceResourcePolicyRequest putRequest =
 PutAccessGrantsInstanceResourcePolicyRequest.builder()
.accountId(111122223333)
.policy(RESOURCE_POLICY)
.build();
PutAccessGrantsInstanceResourcePolicyResponse putResponse =
 s3Control.putAccessGrantsInstanceResourcePolicy(putRequest);
```

```
LOGGER.info("PutAccessGrantsInstanceResourcePolicyResponse: " + putResponse);
}
```

Antwort:

```
PutAccessGrantsInstanceResourcePolicyResponse(
Policy={
"Version": "2012-10-17",
"Statement": [{
"Effect": "Allow",
"Principal": {
"AWS": "444455556666"
},
"Action": [
"s3:ListAccessGrants",
"s3:ListAccessGrantsLocations",
"s3:GetDataAccess"
],
"Resource": "arn:aws:s3:us-east-2:111122223333:access-grants/default"
}]
}
)
```

Example Abrufen einer S3-Access-Grants-Ressourcenrichtlinie

Sie können das folgenden Codebeispiel verwenden, um eine S3-Access-Grants-Ressourcenrichtlinie abzurufen. Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

```
public void getAccessGrantsInstanceResourcePolicy() {
GetAccessGrantsInstanceResourcePolicyRequest getRequest =
 GetAccessGrantsInstanceResourcePolicyRequest.builder()
.accountId(111122223333)
.build();
GetAccessGrantsInstanceResourcePolicyResponse getResponse =
 s3Control.getAccessGrantsInstanceResourcePolicy(getRequest);
LOGGER.info("GetAccessGrantsInstanceResourcePolicyResponse: " + getResponse);
}
```

Antwort:

```
GetAccessGrantsInstanceResourcePolicyResponse(
```

```
Policy={"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":
{"AWS":"arn:aws:iam::444455556666:root"},"Action":
["s3:ListAccessGrants","s3:ListAccessGrantsLocations","s3:GetDataAccess"],"Resource":"arn:aw
east-2:111122223333:access-grants/default"}]},
CreatedAt=2023-06-15T22:54:44.319Z
)
```

Example Löschen einer S3-Access-Grants-Ressourcenrichtlinie

Sie können das folgende Codebeispiel verwenden, um eine S3-Access-Grants-Ressourcenrichtlinie zu löschen. Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

```
public void deleteAccessGrantsInstanceResourcePolicy() {
DeleteAccessGrantsInstanceResourcePolicyRequest deleteRequest =
 DeleteAccessGrantsInstanceResourcePolicyRequest.builder()
.accountId(111122223333)
.build();
DeleteAccessGrantsInstanceResourcePolicyResponse deleteResponse =
 s3Control.putAccessGrantsInstanceResourcePolicy(deleteRequest);
LOGGER.info("DeleteAccessGrantsInstanceResourcePolicyResponse: " + deleteResponse);
}
```

Antwort:

```
DeleteAccessGrantsInstanceResourcePolicyResponse()
```

Verwenden von AWS Tags mit S3 Access Grants

Tags in Amazon S3 Access Grants besitzen ähnliche Eigenschaften wie Objekt-Tags in Amazon S3. Jeder Tag ist ein Schlüssel/Wert-Paar. Die Ressourcen in S3 Access Grants, die Sie markieren können, sind Instances, Speicherorte und Gewährungen.

> **Note**
>
> Beim Markieren in S3 Access Grants werden andere API-Operationen als beim Markieren von Objekten verwendet. S3 Access Grants verwendet die API-Operationen TagResource, UntagResource und ListTagsForResource. Eine Ressource kann eine Instance, ein registrierter Speicherort oder eine Zugriffsgewährung in S3 Access Grants sein.

Ähnlich wie bei Objekt-Tags gelten die folgenden Einschränkungen:

- Sie können Tags zu neuen S3-Access-Grants-Ressourcen hinzufügen, wenn Sie diese erstellen, oder zu vorhandenen Ressourcen hinzufügen.
- Sie können einer Ressource bis zu 10 Tags zuordnen. Wenn mehrere Tags derselben Ressource zugeordnet sind, müssen sie eindeutige Tag-Schlüssel besitzen.
- Ein Tag-Schlüssel kann maximal 128 Unicode-Zeichen lang sein, und die Tag-Werte können bis zu 256 Unicode-Zeichen lang sein. Tags werden intern in UTF-16 dargestellt. In UTF-16 nehmen Zeichen 1 oder 2 Zeichenpositionen ein.
- Schlüssel und Werte unterscheiden nach Groß- und Kleinschreibung.

Weitere Informationen zu Tag-Einschränkungen finden Sie unter Einschränkungen für benutzerdefinierte Tags im AWS Billing -Benutzerhandbuch.

Sie können Ressourcen in S3 Access Grants markieren, indem Sie die AWS Command Line Interface (AWS CLI), die Amazon S3-REST-API oder die AWS SDKs verwenden.

Verwenden der AWS CLI

Informationen zum Installieren der AWS CLI finden Sie unter Installieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

Sie können eine S3-Access-Grants-Ressource während oder nach der Erstellung markieren. Die folgenden Beispiele zeigen, wie Sie eine S3-Access-Grants-Instance markieren oder eine vorhandene Markierung aufheben. Sie können ähnliche Operationen für registrierte Speicherorte und Zugriffsgewährungen durchführen.

Um die folgenden Beispielbefehle zu verwenden, ersetzen Sie *user input placeholders* durch eigene Daten.

Example – Erstellen einer S3-Access-Grants-Instance mit Tags

```
aws s3control create-access-grants-instance \
 --account-id 111122223333 \
 --profile access-grants-profile \
 --region us-east-2 \
 --tags Key=tagKey1,Value=tagValue1
```

Antwort:

```
 {
    "CreatedAt": "2023-10-25T01:09:46.719000+00:00",
    "AccessGrantsInstanceId": "default",
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default"
}
```

Example – Markieren einer bereits erstellten S3-Access-Grants-Instance

```
aws s3control tag-resource \
--account-id 111122223333 \
--resource-arn "arn:aws:s3:us-east-2:111122223333:access-grants/default" \
--profile access-grants-profile \
--region us-east-2 \
--tags Key=tagKey2,Value=tagValue2
```

Example – Auflisten der Tags für eine S3-Access-Grants-Instance

```
aws s3control list-tags-for-resource \
--account-id 111122223333 \
--resource-arn "arn:aws:s3:us-east-2:111122223333:access-grants/default" \
--profile access-grants-profile \
--region us-east-2
```

Antwort:

```
{
    "Tags": [
        {
            "Key": "tagKey1",
            "Value": "tagValue1"
        },
        {
            "Key": "tagKey2",
            "Value": "tagValue2"
        }
    ]
}
```

Example – Aufheben der Markierung für eine S3-Access-Grants-Instance

```
aws s3control untag-resource \
 --account-id 111122223333 \
 --resource-arn "arn:aws:s3:us-east-2:111122223333:access-grants/default" \
 --profile access-grants-profile \
 --region us-east-2 \
 --tag-keys "tagKey2"
```

Verwenden der REST-API

Sie können über die Amazon-S3-API Tags für eine Instance, einen registrierten Standort oder eine Zugriffsgewährung in S3 Access Grants erstellen, entfernen oder auflisten. Informationen zur REST-API-Unterstützung für die Verwaltung von S3-Access-Grants-Tags finden Sie in den folgenden Abschnitten in der Amazon-Simple-Storage-Service-API-Referenz:

- TagResource
- UntagResource
- ListTagsForResource

Limits für S3 Access Grants

Für S3 Access Grants gelten die folgenden Limits:

> **Note**
>
> Wenn Ihr Anwendungsfall diese Einschränkungen überschreitet, wenden Sie sich an den -AWS Support, um höhere Limits anzufordern.

S3-Access-Grants-Instance

Sie können pro AWS-Region Konto 1 S3-Access-Grants-Instance erstellen. Siehe Erstellen einer S3-Access-Grants-Instance.

S3-Access-Grants Speicherort

Sie können pro S3-Access-Grants-Instance 1.000 S3-Access-Grants-Speicherorte registrieren. Siehe Registrieren eines S3-Access-Grants-Speicherorts.

Gewährung

Sie können pro S3-Access-Grants-Instance 100.000 Gewährungen erstellen. Siehe Erstellen einer Gewährung.

S3-Access-Grants-Integrationen

S3 Access Grants kann mit den folgenden AWS Services und Funktionen verwendet werden. Diese Seite wird aktualisiert, wenn neue Integrationen verfügbar sind.

AWS IAM Identity Center

Vertrauenswürdige Identitätsverteilung zwischen Anwendungen

Amazon EMR

Starten eines Amazon-EMR-Clusters mit S3 Access Grants

Amazon EMR in EKS

Starten eines Amazon-EMR-auf-EKS-Clusters mit S3 Access Grants

Amazon-EMR-Serverless-Anwendung

Starten einer Amazon-EMR-Serverless-Anwendung mit S3 Access Grants

Amazon Athena

Verwenden von Athena-Arbeitsgruppen, die für IAM Identity Center aktiviert sind

Zugriffsverwaltung mit ACLs

Zugriffskontrolllisten (ACLs) sind eine der auf Ressourcen basierenden Optionen (siehe Übersicht über die Verwaltung von Zugriffsberechtigungen), mit denen Sie den Zugriff auf Ihre Buckets und Objekte verwalten können. Sie können ACLs verwenden, um anderen grundlegende Lese-/Schreibberechtigungen zu erteilen AWS-Konten. Es gibt Limits für die Verwaltung von Berechtigungen mit ACLs.

Sie können beispielsweise nur anderen Berechtigungen erteilen AWS-Konten. Sie können Benutzern in Ihrem Konto keine Berechtigungen erteilen. Sie können keine bedingten Berechtigung erteilen, und Sie können nicht explizit Berechtigungen verweigern. ACLs sind für spezifische Szenarien geeignet. Wenn ein Bucket-Eigentümer beispielsweise anderen erlaubt, Objekte AWS-Konten hochzuladen,

können Berechtigungen für diese Objekte nur mithilfe von Objekt-ACLs von dem verwaltet werden AWS-Konto , dem das Objekt gehört.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien verwenden, um Zugriff auf Ihren Bucket und die darin enthaltenen Objekte zu gewähren. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Weitere Informationen zu ACLs finden Sie in den folgenden Themen.

Themen

- Zugriffskontrolllisten (ACL) – Übersicht
- Ermitteln der kanonischen Benutzer-ID für Ihr AWS-Konto
- Konfigurieren von ACLs

Zugriffskontrolllisten (ACL) – Übersicht

Amazon-S3-Zugriffskontrolllisten (ACLs) ermöglichen Ihnen die Verwaltung des Bucket- und Objektzugriffs. Jedem Bucket und jedem Objekt ist eine ACL als Subressource zugeordnet. Sie definiert, welche - AWS-Konten oder -Gruppen Zugriff erhalten, und den Zugriffstyp. Wenn eine Anfrage für eine Ressource eingeht, überprüft Amazon S3 die entsprechende ACL,um sicherzustellen, dass der Anforderer die erforderlichen Zugriffsberechtigungen besitzt.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien verwenden, um Zugriff auf Ihren Bucket und die darin enthaltenen Objekte zu gewähren. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Wenn Sie einen Bucket oder ein Objekt erstellen, erstellt Amazon S3 eine Standard-ACL, die dem Ressourcen-Eigentümer die volle Kontrolle über die Ressource erteilt. Dies ist in der folgenden Beispiel-Bucket-ACL gezeigt (die Standard-Objekt-ACL hat denselben Aufbau):

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>*** Owner-Canonical-User-ID ***</ID>
    <DisplayName>owner-display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>*** Owner-Canonical-User-ID ***</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

Die Beispiel-ACL enthält ein `Owner`-Element, das den Eigentümer über die kanonische Benutzer-ID des AWS-Konto identifiziert. Anweisungen zum Auffinden Ihrer kanonischen Benutzer-ID finden Sie unter Eine AWS-Konto kanonische Benutzer-ID finden. Das -`Grant`Element identifiziert den Empfänger (entweder eine AWS-Konto oder eine vordefinierte Gruppe) und die erteilte Berechtigung. Diese Standard-ACL hat ein `Grant`-Element für den Eigentümer. Sie erteilen Berechtigungen, indem Sie `Grant`-Elemente hinzufügen, wobei jedes Recht den Empfänger und die Berechtigung identifiziert.

> **Note**
>
> Eine ACL kann bis zu 100 Rechte haben.

Themen

- Wer ist ein Empfänger?
- Welche Berechtigungen kann ich erteilen?
- aclRequired-Werte für allgemeine Amazon-S3-Anfragen
- Beispiel-ACL
- Vordefinierte ACL

Wer ist ein Empfänger?

Ein Berechtigungsempfänger kann eine AWS-Konto oder eine der vordefinierten Amazon S3-Gruppen sein. Sie erteilen einem AWS-Konto mithilfe der E-Mail-Adresse oder der kanonischen Benutzer-ID die Berechtigung. Wenn Sie jedoch eine E-Mail-Adresse in Ihre Rechteerteilungsanfrage eintragen, findet Amazon S3 die kanonische Benutzer-ID für dieses Konto und fügt sie der ACL hinzu. Die resultierenden ACLs enthalten immer die kanonische Benutzer-ID für das AWS-Konto, nicht die E-Mail-Adresse des AWS-Konto.

Wenn Sie Zugriffsrechte erteilen, geben Sie jeden Empfänger als *type*="*value*"-Paar an, wobei *type* einer der folgenden ist:

- `id` – Wenn der angegebene Wert die kanonische Benutzer-ID eines ist AWS-Konto
- `uri` – Wenn Sie einer vordefinierten Gruppe Berechtigungen erteilen
- `emailAddress` – Wenn der angegebene Wert die E-Mail-Adresse eines AWS-Konto ist

> **Important**
>
> Die Verwendung von E-Mail-Adressen zur Angabe eines Berechtigungsempfängers wird ausschließlich in den folgenden AWS -Regionen unterstützt:
>
> - USA Ost (Nord-Virginia)
> - USA West (Nordkalifornien)
> - USA West (Oregon)
> - Asien-Pazifik (Singapur)
> - Asien-Pazifik (Sydney)
> - Asien-Pazifik (Tokio)
> - Europa (Irland)
> - Südamerika (São Paulo)
>
> Eine Liste aller unterstützten Amazon-S3-Regionen und -Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeine Amazon Web Services-Referenz.

Example Beispiel: E-Mail-Adresse

Der folgende `x-amz-grant-read` Header gewährt beispielsweise dem durch E-Mail-Adressen AWS-Konten identifizierten Berechtigungen zum Lesen von Objektdaten und ihren Metadaten:

```
x-amz-grant-read: emailAddress="xyz@example.com", emailAddress="abc@example.com"
```

> **Warning**
>
> Wenn Sie anderen AWS-Konten Zugriff auf Ihre -Ressourcen gewähren, beachten Sie, dass die ihre Berechtigungen an Benutzer unter ihren Konten delegieren AWS-Konten kann. Man spricht auch von einem kontenübergreifenden Zugriff. Weitere Informationen zum kontenübergreifenden Zugriff finden Sie unter Erstellen einer Rolle, um Berechtigungen an einen IAM-Benutzer zu delegieren im IAM-Benutzerhandbuch.

Eine AWS-Konto kanonische Benutzer-ID finden

Die kanonische Benutzer-ID ist Ihrem AWS-Konto zugeordnet. Diese ID besteht aus einer langen Zeichenfolge, wie z. B.:

`79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be`

Informationen darüber, wo Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Ermitteln der kanonischen Benutzer-ID für Ihr AWS-Konto.

Sie können auch die kanonische Benutzer-ID eines ermitteln, AWS-Konto indem Sie die ACL eines Buckets oder eines Objekts lesen, für das der Zugriffsberechtigungen AWS-Konto hat. Wenn einer Person durch eine Erteilungsanfrage Berechtigungen erteilt AWS-Konto werden, wird der ACL ein Erteilungseintrag mit der kanonischen Benutzer-ID des Kontos hinzugefügt.

> **Note**
>
> Falls Sie Ihren Bucket öffentlich machen (nicht empfohlen), können beliebige, nicht authentifizierte Benutzer Objekte in den Bucket hochladen. Diese anonymen Benutzer haben kein AWS-Konto. Wenn ein anonymer Benutzer ein Objekt in Ihren Bucket hochlädt, fügt Amazon S3 eine spezielle kanonische Benutzer-ID (`65a011a29cdf8ec533ec3d1ccaae921c`) als Objekt-Eigentümer in der ACL hinzu. Weitere Informationen finden Sie unter Amazon-S3-Bucket- und Objekt-Eigentümerschaft.

Vordefinierte Gruppen in Amazon S3

Amazon S3 besitzt mehrere vordefinierte Gruppen. Wenn Sie einem Konto Zugriff auf eine Gruppe erteilen, geben Sie eine der Amazon-S3-URIs statt einer kanonischen Benutzer-ID an. Amazon S3 stellt die folgenden vordefinierten Gruppen bereit:

- Gruppe „Authentifizierte Benutzer“ – Repräsentiert durch `http://acs.amazonaws.com/groups/global/AuthenticatedUsers`.

  Diese Gruppe repräsentiert alle AWS-Konten. Die Zugriffsberechtigung für diese Gruppe erlaubt jedem AWS-Konto , auf die Ressource zuzugreifen. Alle Anfragen müssen jedoch signiert (authentifiziert) sein.

  > **Warning**
  >
  > Wenn Sie Zugriff auf die Gruppe Authentifizierte Benutzer gewähren, kann jeder AWS authentifizierte Benutzer auf der Welt auf Ihre Ressource zugreifen.

- Gruppe „Alle Benutzer“ – Repräsentiert durch `http://acs.amazonaws.com/groups/global/AllUsers`.

  Die Zugriffsberechtigung für diese Gruppe gestattet jedem, auf die Ressource zuzugreifen. Die Anfragen können signiert (authentifiziert) oder nicht signiert (anonym) sein. Nicht signierte Anfragen lassen den Authentifizierungs-Header in der Anfrage weg.

  > **Warning**
  >
  > Wir empfehlen dringend, dass Sie nie der Gruppe Alle Benutzer `WRITE`-, `WRITE_ACP`- oder `FULL_CONTROL`-Berechtigungen erteilen. Während `WRITE`-Berechtigungen es Nichtbesitzern beispielsweise nicht erlauben, vorhandene Objekte zu überschreiben oder zu löschen, erlauben `WRITE`-Berechtigungen jedem, Objekte in Ihrem Bucket zu speichern, für die Ihnen eine Rechnung gestellt wird. Weitere Informationen zu diesen Berechtigungen finden Sie im folgenden Abschnitt Welche Berechtigungen kann ich erteilen?.

- Gruppe „Protokollbereitstellung“ – Repräsentiert durch `http://acs.amazonaws.com/groups/s3/LogDelivery`.

  Die `WRITE`-Berechtigung für einen Bucket gestattet dieser Gruppe, Serverzugriff-Protokolle (siehe Protokollieren von Anfragen mit Server-Zugriffsprotokollierung) in den Bucket zu schreiben.

> **Note**
>
> Bei Verwendung von ACLs kann ein Empfänger eine AWS-Konto oder eine der vordefinierten Amazon S3-Gruppen sein. Der Empfänger kann jedoch kein IAM-Benutzer sein. Weitere Informationen zu AWS -Benutzern und Berechtigungen in IAM finden Sie unter Verwendung von AWS Identity and Access Management.

Welche Berechtigungen kann ich erteilen?

Die folgende Tabelle listet die Berechtigungen auf, die Amazon S3 in einer ACL unterstützt. Die Menge der ACL-Berechtigungen ist für eine Objekt-ACL und eine Bucket-ACL gleich. Abhängig vom Kontext (Bucket-ACL oder Objekt-ACL) erteilen jedoch diese ACL-Berechtigungen die Berechtigungen für spezifische Buckets oder Objekt-Vorgänge. Die Tabelle listet die Berechtigungen auf und beschreibt, was sie im Kontext der Objekte und Buckets bedeuten.

Weitere Informationen zu ACL-Berechtigungen in der Amazon-S3-Konsole finden Sie unter Konfigurieren von ACLs.

ACL-Berechtigungen

Berechtigung	Bei Rechteerteilung für einen Bucket	Bei Rechteerteilung für ein Objekt
`READ`	Gestattet dem Empfänger, die Objekte im Bucket aufzulisten	Gestattet dem Empfänger, die Objektedaten und seine Metadaten zu lesen
`WRITE`	Gestattet dem Empfänger, neue Objekte im Bucket zu erstellen. Für die Bucket- und Objekteigentümer vorhandener Objekte können auch Löschungen und Überschreibungen dieser Objekte ermöglicht werden.	Nicht zutreffend
`READ_ACP`	Gestattet dem Empfänger, die Bucket-ACL zu lesen	Gestattet dem Empfänger, die Objekt-ACL zu lesen
`WRITE_ACP`	Gestattet dem Empfänger, die ACL für den relevanten Bucket zu schreiben	Gestattet dem Empfänger, die ACL für das relevante Objekt zu schreiben

Berechtigung	Bei Rechteerteilung für einen Bucket	Bei Rechteerteilung für ein Objekt
`FULL_CONTROL`	Gewährt dem Berechtigungsempfänger die `READ`-, `WRITE`-, `READ_ACP`- und `WRITE_ACP` - Berechtigungen für den Bucket	Gewährt dem Berechtigungsempfänger die `READ`-, `READ_ACP`- und `WRITE_ACP` -Berechtigungen für das Objekt

> **Warning**
>
> Seien Sie beim Gewähren von Zugriffsberechtigungen auf Ihre S3-Buckets und -Objekte vorsichtig. Beispielsweise kann der Berechtigungsempfänger nach dem Gewähren des `WRITE`-Zugriffs auf einen Bucket Objekte im Bucket erstellen. Wir empfehlen dringend, dass Sie vor dem Erteilen von Berechtigungen den gesamten Abschnitt zu Zugriffskontrolllisten (ACL) – Übersicht lesen.

Mapping der ACL-Berechtigungen und Zugriffsrichtlinienberechtigungen

Wie in der obigen Tabelle gezeigt, erteilt eine ACL nur eine endliche Menge an Berechtigungen im Vergleich zu der Anzahl an Berechtigungen, die Sie in einer Zugriffsrichtlinie festlegen können (siehe Amazon S3-Richtlinienaktionen). Jede dieser Berechtigungen erlaubt einen oder mehrere Amazon-S3-Vorgänge.

Die folgende Tabelle zeigt, wie die verschiedenen ACL-Berechtigungen auf die entsprechenden Zugriffsrichtlinienberechtigungen abgebildet werden. Wie Sie sehen, erteilt die Zugriffsrichtlinie mehr Berechtigungen als eine ACL. Sie verwenden ACLs in erster Linie, um grundlegende Lese-/Schreibberechtigungen zu erteilen, ähnlich den Berechtigungen in einem Dateisystem. Weitere Informationen dazu, wann Sie eine ACL verwenden sollten, finden Sie unter Richtlinien für Zugriffsrichtlinien.

Weitere Informationen zu ACL-Berechtigungen in der Amazon-S3-Konsole finden Sie unter Konfigurieren von ACLs.

ACL-Berechtigung	Entsprechende Zugriffsrichtlinienberechtigungen, wenn einem Bucket die ACL-Berechtigung erteilt wurde	Entsprechende Zugriffsrichtlinienberechtigungen, wenn einem Objekt die ACL-Berechtigung erteilt wurde
`READ`	`s3:ListBucket` , `s3:ListBucketVersions` und `s3:ListBucketMultipartUploads`	`s3:GetObject` und `s3:GetObjectVersion`
`WRITE`	`s3:PutObject` Der Bucket-Eigentümer kann jedes Objekt im Bucket erstellen, überschreiben und löschen, und der Objekteigentümer hat `FULL_CONTROL` über sein Objekt. Wenn der Empfänger der Bucket-Eigentümer ist, gestattet die Erteilung der `WRITE`-Berechtigung in einer Bucket-ACL außerdem, dass die `s3:DeleteObjectVersion` -Aktion für jede Version in diesem Bucket ausgeführt wird.	Nicht zutreffend
`READ_ACP`	`s3:GetBucketAcl`	`s3:GetObjectAcl` und `s3:GetObjectVersionAcl`
`WRITE_ACP`	`s3:PutBucketAcl`	`s3:PutObjectAcl` und `s3:PutObjectVersionAcl`
`FULL_CONTROL`	Dies ist gleichbedeutend mit der Erteilung der `READ`-, `WRITE`-, `READ_ACP`- und `WRITE_ACP` -ACL-Berechtigungen. Dementsprechend wird diese ACL-Berechtigung auf eine Kombination entsprechender Zugriffsrichtlinienberechtigungen abgebildet.	Dies ist gleichbedeutend mit der Erteilung der `READ`-, `READ_ACP`- und `WRITE_ACP` -ACL-Berechtigungen. Dementsprechend wird diese ACL-Berechtigung auf eine Kombination entsprechender Zugriffsrichtlinienberechtigungen abgebildet.

Bedingungsschlüssel

Wenn Sie Zugriffsrichtlinienberechtigungen erteilen, können Sie Bedingungsschlüssel verwenden, um den Wert für die ACL für ein Objekt mithilfe einer Bucket-Richtlinie einzuschränken. Die folgenden Kontextschlüssel entsprechen ACLs. Sie können diese Kontextschlüssel verwenden, um die Verwendung einer bestimmten ACL in einer Anforderung durchzusetzen:

- `s3:x-amz-grant-read` - Erfordert Lesezugriff.
- `s3:x-amz-grant-write` - Erfordert Schreibzugriff.
- `s3:x-amz-grant-read-acp` - Erfordert Lesezugriff auf die Bucket-ACL.
- `s3:x-amz-grant-write-acp` - Erfordert Schreibzugriff auf die Bucket-ACL.
- `s3:x-amz-grant-full-control` - Erfordert vollständige Kontrolle.
- `s3:x-amz-acl` - Erfordert eine Vordefinierte ACL.

Beispielrichtlinien, die ACL-spezifische Header enthalten, finden Sie unter Beispiel 1: Erteilen von s3:PutObject permission mit einer Bedingung, die erfordert, dass der Bucket-Eigentümer die volle Kontrolle erhält. Eine vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

`aclRequired`-Werte für allgemeine Amazon-S3-Anfragen

Wenn Sie Amazon-S3-Anforderungen identifizieren möchten, für die ACLs zur Autorisierung erforderlich waren, können Sie den Wert `aclRequired` in den Amazon-S3-Serverzugriffsprotokollen oder AWS CloudTrail verwenden. Der `aclRequired` Wert, der in den AmazonAmazon S3-Serverzugriffsprotokollen CloudTrail oder angezeigt wird, hängt davon ab, welche Operationen aufgerufen wurden, sowie von bestimmten Informationen über den Anforderer, Objekteigentümer und Bucket-Eigentümer. Wenn keine ACLs erforderlich waren oder wenn Sie die `bucket-owner-full-control` vordefinierte ACL festlegen oder wenn die Anforderungen von Ihrer Bucket-Richtlinie zugelassen werden, lautet die `aclRequired` Wertzeichenfolge in den Amazon S3-Serverzugriffsprotokollen „`-`“ und fehlt in CloudTrail.

In den folgenden Tabellen sind die erwarteten `aclRequired` Werte in den Amazon S3-Serverzugriffsprotokollen CloudTrail oder für die verschiedenen Amazon S3-API-Operationen aufgeführt. Sie können diese Informationen verwenden, um zu verstehen, welche Amazon-S3-Operationen für die Autorisierung von ACLs abhängig sind. In den folgenden Tabellen stellen A, B

und C die verschiedenen Konten dar, die dem Anforderer, Objekteigentümer und Bucket-Besitzer zugeordnet sind. Einträge mit einem Sternchen (*) geben eines der Konten A, B oder C an.

> **Note**
>
> `PutObject`-Operationen in der folgenden Tabelle geben, sofern nicht anders spezifiziert, Anfragen an, die keine ACL festlegen, es sei denn, die ACL ist eine `bucket-owner-full-control`-ACL. Ein Nullwert für `aclRequired` gibt an, dass in AWS CloudTrail Protokollen `aclRequired` fehlt.

`aclRequired` -Werte für CloudTrail

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
`GetObject`	A	A	A	Sie können zwischen Yes und No wählen	Null	Zugriff auf dasselbe Konto
	A	B	A	Sie können zwischen Yes und No wählen	Null	Zugriff auf dasselbe Konto mit erzwungenem Bucket-Eigentümer
	A	A	B	Ja	Null	Von Bucket-Richtlinie gewährter kontoüber greifender Zugriff

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
	A	A	B	Nein	Ja	Kontoübergreifender Zugriff ist auf ACL angewiesen
	A	A	B	Ja	Null	Von Bucket-Richtlinie gewährter kontoübergreifender Zugriff
	A	B	B	Nein	Ja	Kontoübergreifender Zugriff ist auf ACL angewiesen
	A	B	C	Ja	Null	Von Bucket-Richtlinie gewährter kontoübergreifender Zugriff

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
	A	B	C	Nein	Ja	Kontoübergreifender Zugriff ist auf ACL angewiesen
`PutObject`	A	Nicht zutreffend	A	Sie können zwischen Yes und No wählen	Null	Zugriff auf dasselbe Konto
	A	Nicht zutreffend	B	Ja	Null	Von Bucket-Richtlinie gewährter kontoübergreifender Zugriff
	A	Nicht zutreffend	B	Nein	Ja	Kontoübergreifender Zugriff ist auf ACL angewiesen

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
`PutObject` mit einer ACL (außer `bucket-owner-full-control`)	*	Nicht zutreffend	*	Sie können zwischen Yes und No wählen	Ja	Anforderung gewährt ACL
`ListObjects`	A	Nicht zutreffend	A	Sie können zwischen Yes und No wählen	Null	Zugriff auf dasselbe Konto
	A	Nicht zutreffend	B	Ja	Null	Von Bucket-Richtlinie gewährter kontoübergreifender Zugriff
	A	Nicht zutreffend	B	Nein	Ja	Kontoübergreifender Zugriff ist auf ACL angewiesen
`DeleteObject`	A	Nicht zutreffend	A	Sie können zwischen Yes und No wählen	Null	Zugriff auf dasselbe Konto

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
	A	Nicht zutreffend	B	Ja	Null	Von Bucket-Richtlinie gewährter kontoüber greifender Zugriff
	A	Nicht zutreffend	B	Nein	Ja	Kontoüber greifender Zugriff ist auf ACL angewiesen
`PutObjectAcl`	*	*	*	Sie können zwischen Yes und No wählen	Ja	Anforderung gewährt ACL
`PutBucketAcl`	*	Nicht zutreffend	*	Sie können zwischen Yes und No wählen	Ja	Anforderung gewährt ACL

> **Note**
>
> `REST.PUT.OBJECT`-Operationen in der folgenden Tabelle geben, sofern nicht anders spezifiziert, Anfragen an, die keine ACL festlegen, es sei denn, die ACL ist eine `bucket-`

`owner-full-control`-ACL. Eine `aclRequired`-Wertzeichenfolge von „`-`“ gibt einen Nullwert in den Amazon-S3-Serverzugriffsprotokollen an.

`aclRequired`-Werte für Amazon-S3-Server-Zugriffsprotokolle

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
`REST.GET.OBJECT`	A	A	A	Sie können zwischen Yes und No wählen	-	Zugriff auf dasselbe Konto
	A	B	A	Sie können zwischen Yes und No wählen	-	Zugriff auf dasselbe Konto mit erzwungenem Bucket-Eigentümer
	A	A	B	Ja	-	Von Bucket-Richtlinie gewährter kontoübergreifender Zugriff
	A	A	B	Nein	Ja	Kontoübergreifender Zugriff ist auf ACL angewiesen

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
	A	B	B	Ja	-	Von Bucket-Richtlinie gewährter kontoüber greifender Zugriff
	A	B	B	Nein	Ja	Kontoüber greifender Zugriff ist auf ACL angewiesen
	A	B	C	Ja	-	Von Bucket-Richtlinie gewährter kontoüber greifender Zugriff
	A	B	C	Nein	Ja	Kontoüber greifender Zugriff ist auf ACL angewiesen

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
`REST.PUT.OBJECT`	A	Nicht zutreffend	A	Sie können zwischen Yes und No wählen	-	Zugriff auf dasselbe Konto
	A	Nicht zutreffend	B	Ja	-	Von Bucket-Richtlinie gewährter kontoüber greifender Zugriff
	A	Nicht zutreffend	B	Nein	Ja	Kontoüber greifender Zugriff ist auf ACL angewiesen
`REST.PUT.OBJECT` mit einer ACL (außer `bucket-owner-full-control`)	*	Nicht zutreffend	*	Sie können zwischen Yes und No wählen	Ja	Anforderung gewährt ACL
`REST.GET.BUCKET`	A	Nicht zutreffend	A	Sie können zwischen Yes und No wählen	-	Zugriff auf dasselbe Konto

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
	A	Nicht zutreffend	B	Ja	-	Von Bucket-Richtlinie gewährter kontoüber greifender Zugriff
	A	Nicht zutreffend	B	Nein	Ja	Kontoüber greifender Zugriff ist auf ACL angewiesen
`REST.DELETE.OBJECT`	A	Nicht zutreffend	A	Sie können zwischen Yes und No wählen	-	Zugriff auf dasselbe Konto
	A	Nicht zutreffend	B	Ja	-	Von Bucket-Richtlinie gewährter kontoüber greifender Zugriff

Vorgangsname	Auftraggeber	Objekteigentümer	Bucket-Eigentümer	Bucket-Richtlinie gewährt Zugriff	**`aclRequired`** Wert	Grund
	A	Nicht zutreffend	B	Nein	Ja	Kontoübergreifender Zugriff ist auf ACL angewiesen
`REST.PUT.ACL`	*	*	*	Sie können zwischen Yes und No wählen	Ja	Anforderung gewährt ACL

Beispiel-ACL

Die folgende Beispiel-ACL für einen Bucket identifiziert den Ressourcen-Eigentümer und eine Menge von Rechten. Das Format ist die XML-Darstellung einer ACL in der Amazon-S3-REST-API. Der Bucket-Eigentümer hat die `FULL_CONTROL` über die Ressource. Darüber hinaus zeigt die ACL, wie Berechtigungen für eine Ressource an zwei erteilt werden AWS-Konten, identifiziert durch eine kanonische Benutzer-ID, und an zwei der vordefinierten Amazon S3-Gruppen, wie im vorherigen Abschnitt beschrieben.

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>Owner-canonical-user-ID</ID>
    <DisplayName>display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>Owner-canonical-user-ID</ID>
```

```
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>user1-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>WRITE</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>user2-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
      </Grantee>
      <Permission>WRITE</Permission>
    </Grant>

  </AccessControlList>
</AccessControlPolicy>
```

Vordefinierte ACL

Amazon S3 unterstützt einen Satz vordefinierter Rechte, auch als vordefinierte ACLs bezeichnet. Jede vordefinierte ACL hat eine vordefinierte Menge aus Empfängern und Berechtigungen. Die

folgende Tabelle listet die Menge der vordefinierten ACLs und der zugehörigen vordefinierten Rechte auf.

Vordefinierte ACL	Gilt für	Der ACL hinzugefügte Berechtigungen
`private`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL` . Niemand anderer hat Zugriffsrechte (Standard).
`public-read`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL` . Die `AllUsers`-Gruppe (siehe Wer ist ein Empfänger?) erhält `READ`-Zugriff.
`public-read-write`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL` . Die `AllUsers`-Gruppe erhält `READ`- und `WRITE`-Zugriff. Eine solche Erteilung von Rechten für einen Bucket wird im Allgemeinen nicht empfohlen.
`aws-exec-read`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL` . Amazon EC2; erhält `GET`-Zugriff auf ein `READ`, ein Amazon Machine Image (AMI)-Paket von Amazon S3.
`authenticated-read`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL` . Die `AuthenticatedUsers` -Gruppe erhält `READ`-Zugriff.
`bucket-owner-read`	Objekt	Der Objekt-Eigentümer erhält `FULL_CONTROL` . Der Bucket-Eigentümer erhält `READ`-Zugriff. Wenn Sie diese vordefinierte ACL beim Erstellen eines Buckets angeben, ignoriert Amazon S3 sie.
`bucket-owner-full-control`	Objekt	Sowohl der Objekt-Eigentümer, als auch der Bucket-Eigentümer erhalten `FULL_CONTROL` für das Objekt. Wenn Sie diese vordefinierte ACL beim Erstellen eines Buckets angeben, ignoriert Amazon S3 sie.
`log-delivery-write`	Bucket	Die `LogDelivery` -Gruppe erhält `WRITE`- und `READ_ACP`-Berechtigungen für den Bucket. Weitere

Vordefinierte ACL	Gilt für	Der ACL hinzugefügte Berechtigungen
		Informationen über Protokolle finden Sie unter (Protokollieren von Anfragen mit Server-Zugriffsprotokollierung).

> **Note**
>
> Sie können auch in Ihrer Anfrage nur eine dieser vordefinierten ACLs angeben.

Sie geben mit dem Anfrage-Header `x-amz-acl` eine vordefinierte ACL in Ihrer Anfrage an. Wenn Amazon S3 eine Anfrage mit einer vordefinierten ACL erhält, fügt es die vordefinierten Rechte der ACL der Ressource hinzu.

Ermitteln der kanonischen Benutzer-ID für Ihr AWS-Konto

Die kanonische Benutzer-ID ist eine alphanumerische Kennung, z. B. `79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be`, die eine verschleierte Form der AWS-Konto ID darstellt. Sie können diese ID verwenden, um ein AWS-Konto zu identifizieren, wenn Sie mit Amazon S3 den kontoübergreifenden Zugriff auf Buckets und Objekte gewähren. Die kanonische Benutzer-ID für Ihr Konto können Sie als Stammbenutzer, verbundener Benutzer oder IAM-Benutzer abrufen.

Sie finden die kanonische Benutzer-ID für Ihr AWS-Konto mithilfe der AWS Management Console oder der AWS CLI. Die kanonische Benutzer-ID für einen AWS-Konto ist für dieses Konto spezifisch.

Voraussetzungen

Wenn Sie ein Verbundbenutzer sind oder programmgesteuert auf die Informationen zugreifen, z. B. über die AWS CLI, müssen Sie über die Berechtigung zum Auflisten und Anzeigen eines Amazon S3-Buckets verfügen.

Verwenden der S3-Konsole (Stammbenutzer oder IAM-Benutzer)

Gehen Sie wie folgt vor, um die kanonische Benutzer-ID für Ihr zu finden AWS-Konto , wenn Sie als Stammbenutzer oder IAM-Benutzer bei der Konsole angemeldet sind. Weitere Informationen über Stammbenutzer und IAM-Benutzer finden Sie unter Übersicht über die AWS -Identitätsverwaltung: Benutzer im IAM-Benutzerhandbuch.

1. Melden Sie sich als Stammbenutzer oder IAM-Benutzer bei der Konsole an.

   Weitere Informationen finden Sie unter Bei der AWS Management Console anmelden) im IAM-Benutzerhandbuch.
2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Kontonamen oder Ihre Kontonummer und dann Security Credentials (Meine Sicherheitsanmeldeinformationen) aus.
3. Ermitteln Sie die kanonische ID für das Konto:
   - Wenn Sie der Stammbenutzer sind, dann erweitern Sie Konto-Kennungen und suchen Sie die kanonische Benutzer-ID.
   - Wenn Sie ein IAM-Benutzer sind, dann finden Sie die kanonische Benutzer-ID des Kontos unter Kontodetails.

Verwenden der S3-Konsole (verbundener Benutzer)

Gehen Sie wie folgt vor, um die kanonische Benutzer-ID für Ihr Konto zu finden, wenn Sie bei der AWS Management Console als Verbundbenutzer angemeldet sind. Weitere Informationen zu verbundenen Benutzern finden Sie unter Erstellen eines Verbunds von vorhandenen Benutzern im IAM-Benutzerhandbuch.

> **Note**
>
> Um zu überprüfen, ob Sie sich bei der AWS Management Console als Verbundbenutzer angemeldet haben, wählen Sie auf der AWS Management Console Seite Ihre Kontoinformationen aus und überprüfen Sie die erweiterten Kontoinformationen. Wenn in den Kontoinformationen Federated user (Verbundbenutzer) angezeigt wird, sind Sie als Verbundbenutzer angemeldet.

1. Melden Sie sich bei der Konsole als verbundener Benutzer an.

   Weitere Informationen finden Sie unter Signing in to the AWS Management Console(Bei der Konsole anmelden) im IAM-Benutzerhandbuch.
2. Wählen Sie in der Amazon-S3-Konsole einen Bucket-Namen aus, um die Bucket-Details anzuzeigen.
3. Klicken Sie auf Permissions (Berechtigungen) und scrollen Sie dann nach unten zum Abschnitt Access control list (ACL) (Zugriffssteuerungsliste (ACL)).

Unter Bucket-Eigentümer (Ihr AWS Konto) wird die kanonische Benutzer-ID für die AWS-Konto angezeigt.

Verwenden der AWS CLI

Verwenden Sie den Befehl list-buckets wie folgt, um die kanonische Benutzer-ID mithilfe der AWS CLI zu finden.

```
aws s3api list-buckets --query Owner.ID --output text
```

Konfigurieren von ACLs

In diesem Abschnitt wird die Verwaltung von Zugriffsberechtigungen für S3-Buckets und Objekten unter Verwendung von Zugriffskontrolllisten (ACLs) beschrieben. Sie können Ihrer Ressourcen-ACL mithilfe der AWS Management Console, der AWS Command Line Interface (CLI), der REST-API oder AWS SDKs Erteilungen hinzufügen.

Bucket- und Objekt-Berechtigungen sind voneinander unabhängig. Ein Objekt erbt nicht die Berechtigungen von seinem Bucket. Wenn Sie beispielsweise einen Bucket erstellen und einem Benutzer Schreibzugriff erteilen, können Sie auf die Objekte dieses Benutzers nicht zugreifen, wenn Ihnen der Benutzer nicht explizit Zugriff erteilt.

Sie können anderen AWS-Konto Benutzern oder vordefinierten Gruppen Berechtigungen erteilen. Der Benutzer oder die Gruppe, dem bzw. der Sie Berechtigungen erteilen, wird als der Berechtigungsempfänger bezeichnet. Standardmäßig hat der Eigentümer, nämlich das AWS-Konto , das den Bucket erstellt hat, alle Berechtigungen.

Für jede Berechtigung, die Sie für einen Benutzer oder eine Gruppe erteilen, wird der dem Bucket zugeordneten ACL ein Eintrag hinzugefügt. Die ACL listet die erteilten Berechtigungen auf, die den Berechtigungsempfänger und die erteilte Berechtigung identifizieren.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Important

Wenn Ihr Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien verwenden, um Zugriff auf Ihren Bucket und die darin enthaltenen Objekte zu gewähren. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Warning

Es wird dringend empfohlen, Schreibzugriff für die Gruppen Jeder (öffentlicher Zugriff) oder Authentifizierte Benutzer (alle AWS authentifizierten Benutzer) zu gewähren. Weitere Informationen zu den Auswirkungen eines Schreibzugriffs für diese Gruppen finden Sie unter Vordefinierte Gruppen in Amazon S3.

Festlegen von ACL-Berechtigungen für einen Bucket mit der S3-Konsole

Die Konsole zeigt kombinierte Zugriffsberechtigungen für doppelte Berechtigungsempfänger an. Um die vollständige Liste der ACLs anzuzeigen, verwenden Sie die Amazon S3-REST-API AWS CLI oder - AWS SDKs.

Die folgende Tabelle zeigt die ACL-Berechtigungen, die Sie für Buckets in der Amazon-S3-Konsole konfigurieren können.

Amazon-S3-Konsolen-ACL-Berechtigungen für Buckets

Konsolenberechtigung	ACL-Berechtigung	Zugriff
Objekte – Auflisten	`READ`	Gestattet dem Empfänger, die Objekte im Bucket aufzulisten.
Objekte – Schreiben	`WRITE`	Gestattet dem Empfänger, neue Objekte im Bucket zu erstellen. Für die Bucket- und Objekteigentümer vorhandener Objekte können auch Löschungen und Überschreibungen dieser Objekte ermöglicht werden.
Bucket-ACL – Lesen	`READ_ACP`	Gestattet dem Empfänger, die Bucket-ACL zu lesen.
Bucket-ACL – Schreiben	`WRITE_ACP`	Gestattet dem Empfänger, die ACL für den relevanten Bucket zu schreiben.
Jeder (öffentlicher Zugang): Objekte – Auflisten	`READ`	Gewährt öffentlichen Lesezugriff für die Objekte im Bucket. Wenn Sie jedem (öffentlichen Zugriff) Listenzugriff gewähren, kann jeder Benutzer auf der Welt auf die Objekte im Bucket zugreifen.
Jeder (öffentlicher Zugriff): Bucket-ACL – Lesen	`READ_ACP`	Gewährt öffentlichen Lesezugriff für die Bucket-ACL. Wenn Sie jedem (öffentlicher Zugriff) Lesezugriff gewähren, kann jeder Benutzer auf der Welt auf die Bucket-ACL zugreifen.

Weitere Informationen zu ACL-Berechtigungen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.

Important

Wenn Ihr Bucket die Einstellung „Vom Bucket-Besitzer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien für die Gewährung des Zugriffs auf Ihren Bucket und die enthaltenen Objekte verwenden. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von

Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Festlegen von ACL-Berechtigungen für einen Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie Berechtigungen festlegen möchten.
3. Wählen Sie Permissions (Berechtigungen).
4. Wählen Sie unter Access Control List Bearbeiten.

   Sie können die folgenden ACL-Berechtigungen für den Bucket bearbeiten:

   Objekte

   - Auflisten – Gestattet einem Empfänger, die Objekte im Bucket aufzulisten.
   - Schreiben — Ermöglicht es dem Berechtigungsempfänger, neue Objekte im Bucket zu erstellen. Für die Bucket- und Objekteigentümer vorhandener Objekte können auch Löschungen und Überschreibungen dieser Objekte ermöglicht werden.

     In der S3-Konsole können Sie nur Schreibzugriff auf die S3-Protokollbereitstellungsgruppe und den Bucket-Eigentümer (Ihr AWS-Konto) gewähren. Wir empfehlen dringend, anderen Empfängern keinen Schreibzugriff zu gewähren. Wenn Sie jedoch Schreibzugriff gewähren müssen, können Sie die AWS CLI, - AWS SDKs oder die REST-API verwenden.

   Bucket-ACL

   - Lesen – Gestattet dem Empfänger, die Bucket-ACL zu lesen.
   - Schreiben – Gestattet dem Empfänger, die ACL für den relevanten Bucket zu schreiben.
5. Um die Berechtigungen des Bucket-Eigentümers zu ändern, löschen oder wählen Sie neben dem Bucket-Eigentümer (Ihr AWS-Konto) die folgenden ACL-Berechtigungen aus:

   - Objekte – Auflisten oder Schreiben
   - Bucket-ACL – Lesen oder Schreiben

Der Eigentümer bezieht sich auf die Root-Benutzer des AWS-Kontos, nicht auf einen AWS Identity and Access Management IAM-Benutzer. Weitere Informationen zum Root-Benutzer finden Sie unter Der Root-Benutzer des AWS-Kontos im IAM-Benutzerhandbuch.

6. Um Berechtigungen für die Allgemeinheit (alle im Internet) zu erteilen oder rückgängig zu machen, deaktivieren oder wählen Sie neben Jeder (öffentlicher Zugriff) die folgenden ACL-Berechtigungen aus:

   - Objekte – Auflisten
   - Bucket-ACL – Lesen

   > **Warning**
   >
   > Seien Sie vorsichtig, wenn Sie der Gruppe Everyone (Jeder) öffentlichen Zugriff auf Ihren S3-Bucket gewähren. Wenn Sie dieser Gruppe anonymen Zugriff gewähren, kann jeder auf der ganzen Welt auf Ihren Bucket zugreifen. Wir empfehlen dringend, nie einen öffentlichen Schreibzugriff auf Ihren S3-Bucket zu gewähren.

7. Um Berechtigungen für Personen mit einem zu erteilen oder rückgängig zu machen AWS-Konto, löschen oder wählen Sie neben der Gruppe Authentifizierte Benutzer (alle Benutzer mit einem AWS-Konto) die folgenden ACL-Berechtigungen aus:

   - Objekte – Auflisten
   - Bucket-ACL – Lesen

8. Um Amazon S3 Berechtigungen zum Schreiben von Server-Zugriffsprotokollen in den Bucket zu erteilen oder rückgängig zu machen, deaktivieren oder wählen Sie unter der S3-Protokoll-Bereitstellungsgruppe die folgenden ACL-Berechtigungen aus:

   - Objekte – Auflisten oder Schreiben
   - Bucket-ACL – Lesen oder Schreiben

   Wenn ein Bucket als Ziel-Bucket für Zugriffsprotokolle eingerichtet ist, müssen die Bucket-Berechtigungen der Gruppe Log Delivery (Protokollbereitstellung) Schreibzugriff auf den Bucket erteilen. Wenn Sie die Server-Zugriffsprotokollierung für einen Bucket aktivieren, erteilt die Amazon-S3-Konsole der Gruppe Log Delivery (Protokollbereitstellung) Schreibzugriff auf den Ziel-Bucket, den Sie für den Empfang der Protokolle ausgewählt haben. Weitere

Informationen zu Server-Zugriffsprotokollen finden Sie unter Aktivieren Sie die Amazon-S3-Server-Zugriffsprotokollierung.

9. Gehen Sie wie folgt vor AWS-Konto, um Zugriff auf ein anderes zu gewähren:

   a. Wählen Sie Empfänger hinzufügen.
   b. Geben Sie im Feld Empfänger die kanonische ID des anderen AWS-Konto ein.
   c. Wählen Sie aus den folgenden ACL-Berechtigungen aus:

      - Objekte – Auflisten oder Schreiben
      - Bucket-ACL – Lesen oder Schreiben

   > **Warning**
   >
   > Wenn Sie anderen AWS-Konten Zugriff auf Ihre -Ressourcen gewähren, beachten Sie, dass die ihre Berechtigungen an Benutzer unter ihren Konten delegieren AWS-Konten kann. Man spricht auch von einem kontenübergreifenden Zugriff. Weitere Informationen zum kontenübergreifenden Zugriff finden Sie unter Erstellen einer Rolle, um Berechtigungen an einen IAM-Benutzer zu delegieren im IAM-Benutzerhandbuch.

10. Um den Zugriff auf ein anderes zu entfernen AWS-Konto, wählen Sie unter Zugriff für andere die AWS-Konten Option Entfernen aus.
11. Klicken Sie auf Save changes (Änderungen speichern), um die Änderungen zu speichern.

Festlegen von ACL-Berechtigungen für ein Objekt mit der S3-Konsole

Die Konsole zeigt kombinierte Zugriffsberechtigungen für doppelte Berechtigungsempfänger an. Um die vollständige Liste der ACLs anzuzeigen, verwenden Sie die Amazon S3-REST-API AWS CLI oder - AWS SDKs. Die folgende Tabelle zeigt die ACL-Berechtigungen, die Sie für Objekte in der Amazon-S3-Konsole konfigurieren können.

Amazon-S3-Konsolen-ACL-Berechtigungen für Objekte

Konsolenberechtigung	ACL-Berechtigung	Zugriff
Objekt – Lesen	`READ`	Gestattet dem Empfänger, die Objektedaten und seine Metadaten zu lesen.

Konsolenberechtigung	ACL-Berechtigung	Zugriff
Objekt-ACL – Lesen	`READ_ACP`	Gestattet dem Empfänger, die Objekt-ACL zu lesen.
Objekt-ACL – Schreiben	`WRITE_ACP`	Gestattet dem Empfänger, die ACL für das relevante Objekt zu schreiben

Weitere Informationen zu ACL-Berechtigungen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Besitzer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien für die Gewährung des Zugriffs auf Ihren Bucket und die enthaltenen Objekte verwenden. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

ACL-Berechtigungen für ein Objekt festlegen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
3. Wählen Sie in der Liste Objekte den Namen des Objekts aus, für das Sie Berechtigungen festlegen möchten.
4. Wählen Sie Permissions (Berechtigungen).
5. Wählen Sie unter Access Control List (ACL) die Option Bearbeiten.

   Sie können die folgenden ACL-Berechtigungen für das Objekt bearbeiten:

Object

- Lesen – Gestattet dem Empfänger, die Objektedaten und seine Metadaten zu lesen.

Objekt-ACL

- Lesen – Gestattet dem Empfänger, die Objekt-ACL zu lesen.
- Schreiben – Gestattet dem Empfänger, die ACL für das relevante Objekt zu schreiben. In der S3-Konsole können Sie nur dem Bucket-Eigentümer (Ihrem) Schreibzugriff gewähren AWS-Konto. Wir empfehlen dringend, anderen Empfängern keinen Schreibzugriff zu gewähren. Wenn Sie jedoch Schreibzugriff gewähren müssen, können Sie die AWS CLI, AWS SDKs oder die REST-API verwenden.

6. Sie können Zugriffsberechtigungen von Objekten für Folgendes verwalten:

   a. Zugriff für den Besitzer des Objekts

      Der Eigentümer bezieht sich auf die Root-Benutzer des AWS-Kontos und nicht auf einen AWS Identity and Access Management IAM-Benutzer. Weitere Informationen zum Root-Benutzer finden Sie unter Der Root-Benutzer des AWS-Kontos im IAM-Benutzerhandbuch.

      Um die Objektzugriffsberechtigungen des Besitzers zu ändern, wählen Sie unter Zugriff für Objekteigentümer die Option Ihr AWS Konto (Besitzer) aus.

      Aktivieren Sie die Kontrollkästchen für die Berechtigungen, die geändert werden sollen. Wählen Sie dann Save (Speichern) aus.

   b. Zugriff für andere AWS-Konten

      Um einem - AWS Benutzer aus einem anderen Berechtigungen zu erteilen AWS-Konto, wählen Sie unter Zugriff für andere die AWS-Konten Option Konto hinzufügen aus. Geben Sie im Feld ID eingeben die kanonische ID des AWS Benutzers ein, dem Sie Objektberechtigungen erteilen möchten. Informationen zur Suche nach einer kanonischen ID finden Sie unter Ihre AWS-Konto -Kennungen in der Allgemeine Amazon Web Services-Referenz. Sie können bis zu 99 Benutzer hinzufügen.

      Aktivieren Sie die Kontrollkästchen für die Berechtigungen, die dem Benutzer gewährt werden sollen. Wählen Sie dann Save (Speichern) aus. Um Informationen über die Berechtigungen anzuzeigen, wählen Sie die Hilfesymbole aus.

c. Öffentlicher Zugriff

Um der allgemeinen Öffentlichkeit (jedem Benutzer auf der ganzen Welt) Zugriff auf Ihr Objekt zu gewähren, wählen Sie unter Public access (Öffentlicher Zugriff) Everyone (Jeder) aus. Die Erteilung öffentlicher Zugriffsberechtigungen bedeutet, dass jeder Benutzer auf der ganzen Welt auf das Objekt zugreifen kann.

Aktivieren Sie die Kontrollkästchen für die Berechtigungen, die erteilt werden sollen. Wählen Sie dann Save (Speichern) aus.

> **Warning**
>
> - Seien Sie vorsichtig, wenn Sie der Gruppe Everyone (Jeder) anonymen Zugriff auf Ihre Amazon-S3-Objekte gewähren. Wenn Sie dieser Gruppe Zugriff gewähren, kann jeder auf der ganzen Welt auf Ihr Objekt zugreifen. Wenn Sie jedem Zugriff gewähren müssen, wird nachdrücklich empfohlen, dass Sie nur Berechtigungen für Read objects (Objekte lesen) gewähren.
> - Wir empfehlen nachdrücklich, der Gruppe Everyone (Jeder) keine Schreibberechtigungen für Objekte zu erteilen. Dadurch können alle Benutzer die ACL-Berechtigungen für das Objekt überschreiben.

Verwenden der AWS SDKs

Dieser Abschnitt enthält Beispiele, wie Access-Control-List(ACL)-Berechtigungen für Buckets und Objekte konfiguriert werden.

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Besitzer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien für die Gewährung des Zugriffs auf Ihren Bucket und die enthaltenen Objekte verwenden. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Java

Dieser Abschnitt enthält Beispiele, wie Access-Control-List(ACL)-Berechtigungen für Buckets und Objekte konfiguriert werden. Das erste Beispiel erstellt einen Bucket mit einer vorgefertigten ACL (siehe Vordefinierte ACL), erstellt eine Liste benutzerdefinierter Berechtigungen und ersetzt die vorgefertigte ACL durch eine ACL mit den benutzerdefinierten Berechtigungen. Das zweite Beispiel veranschaulicht, wie Sie eine ACL mit der Methode `AccessControlList.grantPermission()` abändern.

Example Erstellen Sie einen Bucket und geben Sie eine vordefinierte ACL an, die der S3-Protokoll-Bereitstellungsgruppe die Berechtigung erteilt

Dieses Beispiel erstellt einen Bucket. In der Anfrage gibt das Beispiel eine vorgefertigte ACL an, die die Protokoll-Bereitstellungsgruppe zum Schreiben von Protokollen in den Bucket berechtigt.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.ArrayList;

public class CreateBucketWithACL {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String userEmailForReadPermission = "*** user@example.com ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .build();

            // Create a bucket with a canned ACL. This ACL will be replaced by the
            // setBucketAcl()
            // calls below. It is included here for demonstration purposes.
            CreateBucketRequest createBucketRequest = new
 CreateBucketRequest(bucketName, clientRegion.getName())
```

```
                    .withCannedAcl(CannedAccessControlList.LogDeliveryWrite);
            s3Client.createBucket(createBucketRequest);

            // Create a collection of grants to add to the bucket.
            ArrayList<Grant> grantCollection = new ArrayList<Grant>();

            // Grant the account owner full control.
            Grant grant1 = new Grant(new
 CanonicalGrantee(s3Client.getS3AccountOwner().getId()),
                    Permission.FullControl);
            grantCollection.add(grant1);

            // Grant the LogDelivery group permission to write to the bucket.
            Grant grant2 = new Grant(GroupGrantee.LogDelivery, Permission.Write);
            grantCollection.add(grant2);

            // Save grants by replacing all current ACL grants with the two we just
 created.
            AccessControlList bucketAcl = new AccessControlList();
            bucketAcl.grantAllPermissions(grantCollection.toArray(new Grant[0]));
            s3Client.setBucketAcl(bucketName, bucketAcl);

            // Retrieve the bucket's ACL, add another grant, and then save the new
 ACL.
            AccessControlList newBucketAcl = s3Client.getBucketAcl(bucketName);
            Grant grant3 = new Grant(new
 EmailAddressGrantee(userEmailForReadPermission), Permission.Read);
            newBucketAcl.grantAllPermissions(grant3);
            s3Client.setBucketAcl(bucketName, newBucketAcl);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Aktualisieren der ACL für ein bestehendes Objekt

Dieses Beispiel aktualisiert die ACL für ein Objekt. Das Beispiel führt die folgenden Aufgaben durch:

- Ruft die ACL eines Objekts ab
- Löscht die ACL durch Entfernen aller vorhandenen Berechtigungen
- Fügt zwei Berechtigungen hinzu: Vollzugriff für den Eigentümer und WRITE_ACP (siehe Welche Berechtigungen kann ich erteilen?) für einen anhand einer E-Mail-Adresse identifizierten Benutzer
- Speichert die ACL im Objekt

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AccessControlList;
import com.amazonaws.services.s3.model.CanonicalGrantee;
import com.amazonaws.services.s3.model.EmailAddressGrantee;
import com.amazonaws.services.s3.model.Permission;

import java.io.IOException;

public class ModifyACLExistingObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String emailGrantee = "*** user@example.com ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Get the existing object ACL that we want to modify.
```

```
            AccessControlList acl = s3Client.getObjectAcl(bucketName, keyName);

            // Clear the existing list of grants.
            acl.getGrantsAsList().clear();

            // Grant a sample set of permissions, using the existing ACL owner for
 Full
            // Control permissions.
            acl.grantPermission(new CanonicalGrantee(acl.getOwner().getId()),
 Permission.FullControl);
            acl.grantPermission(new EmailAddressGrantee(emailGrantee),
 Permission.WriteAcp);

            // Save the modified ACL back to the object.
            s3Client.setObjectAcl(bucketName, keyName, acl);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Example Erstellen Sie einen Bucket und geben Sie eine vordefinierte ACL an, die der S3-Protokoll-Bereitstellungsgruppe die Berechtigung erteilt

Dieses C#-Beispiel erstellt einen Bucket. In der Anfrage gibt der Code auch eine vorgefertigte ACL an, die die Protokoll-Bereitstellungsgruppe zum Schreiben der Protokolle in den Bucket berechtigt.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
```

```
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ManagingBucketACLTest
    {
        private const string newBucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            CreateBucketUseCannedACLAsync().Wait();
        }

        private static async Task CreateBucketUseCannedACLAsync()
        {
            try
            {
                // Add bucket (specify canned ACL).
                PutBucketRequest putBucketRequest = new PutBucketRequest()
                {
                    BucketName = newBucketName,
                    BucketRegion = S3Region.EUW1, // S3Region.US,
                                                  // Add canned ACL.
                    CannedACL = S3CannedACL.LogDeliveryWrite
                };
                PutBucketResponse putBucketResponse = await
 client.PutBucketAsync(putBucketRequest);

                // Retrieve bucket ACL.
                GetACLResponse getACLResponse = await client.GetACLAsync(new
 GetACLRequest
                {
                    BucketName = newBucketName
                });
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
```

```
                Console.WriteLine("S3 error occurred. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Example Aktualisieren der ACL für ein bestehendes Objekt

Dieses C#-Beispiel aktualisiert die ACL für ein vorhandenes Objekt. Das Beispiel führt die folgenden Aufgaben durch:

- Ruft die ACL eines Objekts ab.
- Löscht die ACL durch Entfernen aller vorhandenen Berechtigungen.
- Fügt zwei Berechtigungen hinzu: Vollzugriff für den Eigentümer und WRITE_ACP für einen anhand einer E-Mail-Adresse identifizierten Benutzer.
- Speichert die ACL durch Senden einer `PutAcl`-Anfrage.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ManagingObjectACLTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** object key name ***";
        private const string emailAddress = "*** email address ***";
        // Specify your bucket region (an example region is shown).
```

```
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            TestObjectACLTestAsync().Wait();
        }
        private static async Task TestObjectACLTestAsync()
        {
            try
            {
                    // Retrieve the ACL for the object.
                    GetACLResponse aclResponse = await client.GetACLAsync(new
GetACLRequest
                    {
                        BucketName = bucketName,
                        Key = keyName
                    });

                    S3AccessControlList acl = aclResponse.AccessControlList;

                    // Retrieve the owner (we use this to re-add permissions after
we clear the ACL).
                    Owner owner = acl.Owner;

                    // Clear existing grants.
                    acl.Grants.Clear();

                    // Add a grant to reset the owner's full permission (the
previous clear statement removed all permissions).
                    S3Grant fullControlGrant = new S3Grant
                    {
                        Grantee = new S3Grantee { CanonicalUser = owner.Id },
                        Permission = S3Permission.FULL_CONTROL

                    };

                    // Describe the grant for the permission using an email address.
                    S3Grant grantUsingEmail = new S3Grant
                    {
                        Grantee = new S3Grantee { EmailAddress = emailAddress },
                        Permission = S3Permission.WRITE_ACP
                    };
```

```
                    acl.Grants.AddRange(new List<S3Grant> { fullControlGrant,
 grantUsingEmail });

                    // Set a new ACL.
                    PutACLResponse response = await client.PutACLAsync(new
 PutACLRequest
                    {
                        BucketName = bucketName,
                        Key = keyName,
                        AccessControlList = acl
                    });
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Verwenden der REST-API

Amazon-S3-APIs ermöglichen Ihnen, beim Erstellen eines Buckets oder eines Objekts eine ACL einzurichten. Amazon S3 bietet auch eine API für die Einrichtung einer ACL für einen vorhandenen Bucket oder ein Objekt. Diese APIs stellen die folgenden Methoden bereit, um eine ACL einzurichten:

- Einrichten von ACL mit Anfrageheadern – Wenn Sie eine Anfrage senden, um eine Ressource zu erstellen (Bucket oder Objekt), richten Sie eine ACL über die Anfrage-Header ein. Mit Hilfe dieser Header können Sie entweder eine vordefinierte ACL angeben, oder explizit Rechte erteilen (mit expliziter Identifizierung von Empfänger und Berechtigungen).
- Einrichtung von ACL mithilfe des Anfragetextes – Wenn Sie eine Anfrage senden, um ACL für eine vorhandene Ressource einzurichten, können Sie die ACL im Header oder im Text der Anfrage einrichten.

Informationen zur REST-API-Unterstützung für die Verwaltung von ACLs finden Sie in den folgenden Abschnitten der Amazon Simple Storage Service API-Referenz:

- GET Bucket ACL
- PUT Bucket ACL
- GET Object acl
- PUT Object ACL
- PUT Object
- PUT Bucket
- PUT Object – Kopieren
- Initiieren eines mehrteiligen Uploads

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Besitzer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien für die Gewährung des Zugriffs auf Ihren Bucket und die enthaltenen Objekte verwenden. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Spezifische Anforderungs-Header für Access Control List (ACL)

Sie können Header verwenden, um Berechtigungen auf der Basis von Access Control List (ACL) zu erteilen. Standardmäßig sind alle Objekte privat. Nur der Besitzer hat die vollständige Zugriffssteuerung. Wenn Sie ein neues Objekt hinzufügen, können Sie einzelnen AWS-Konten oder vordefinierten Gruppen, die von Amazon S3 definiert werden, Berechtigungen erteilen. Diese Berechtigungen werden anschließend der Zugriffskontrollliste (Access Control List, ACL) für das Objekt hinzugefügt. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.

Mit dieser Operation können Sie Zugriffsberechtigungen mit einer der folgenden beiden Methoden erteilen:

- Vordefinierte ACL (**`x-amz-acl`**) – Amazon S3 unterstützt einen Satz vordefinierter ACLs, englisch „Canned ACLs“. Jede vordefinierte ACL hat eine vordefinierte Menge aus Empfängern und Berechtigungen. Weitere Informationen finden Sie unter Vordefinierte ACL.
- Zugriffsberechtigungen – Um bestimmten AWS-Konten oder Gruppen explizit Zugriffsberechtigungen zu erteilen, verwenden Sie die folgenden Header. Jeder Header ist bestimmten Berechtigungen zugeordnet, die Amazon S3 in einer ACL unterstützt. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht. Im Header geben Sie eine Liste der Empfänger an, die die jeweilige Berechtigung erhalten.
  - x-amz-grant-read
  - x-amz-grant-write
  - x-amz-grant-read-acp
  - x-amz-grant-write-acp
  - x-amz-grant-full-Kontrolle

Verwenden der AWS CLI

Weitere Informationen zum Verwalten von ACLs mit der finden Sie AWS CLI unter put-bucket-acl in der AWS CLI -Befehlsreferenz.

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien verwenden, um Zugriff auf Ihren Bucket und die darin enthaltenen Objekte zu gewähren. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Cross-Origin Resource Sharing (CORS) verwenden

Cross-Origin Resource Sharing (CORS) bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain. Mit CORS-Unterstützung können Sie umfassende clientseitige Webanwendungen mit Amazon S3 erstellen und selektiven ursprungsübergreifenden Zugriff auf Ihre Amazon-S3-Ressourcen zulassen.

Dieser Abschnitt bietet eine Übersicht über CORS. In den Unterthemen wird beschrieben, wie Sie CORS über die Amazon S3-Konsole oder programmgesteuert über die Amazon S3-REST-API und die AWS SDKs aktivieren können.

Cross-Origin Resource Sharing: Szenarien in Anwendungsfällen

Es folgen typische Beispielszenarien für den Einsatz von CORS.

Szenario 1

Szenario 1: Angenommen, Sie hosten eine Website in einem Amazon-S3-Bucket mit dem Namen `website`, wie in Hosten einer statischen Website mit Amazon S3 beschrieben. Ihre Benutzer laden den Website-Endpunkt:

```
http://website.s3-website.us-east-1.amazonaws.com
```

Jetzt möchten Sie JavaScript auf den Webseiten verwenden, die in diesem Bucket gespeichert sind, um authentifizierte GET- und PUT-Anfragen für denselben Bucket unter Verwendung des Amazon S3-API-Endpunkts für den Bucket senden zu können, `website.s3.us-east-1.amazonaws.com`. Ein Browser würde normalerweise JavaScript daran hindern, diese Anfragen zuzulassen, aber mit CORS können Sie Ihren Bucket so konfigurieren, dass er explizit ursprungsübergreifende Anfragen von aktiviert`website.s3-website.us-east-1.amazonaws.com`.

Szenario 2

Angenommen, Sie möchten eine Web-Schriftart aus Ihrem S3-Bucket hosten. Auch hier erfordern Browser eine CORS-Prüfung (auch als Preflight-Check bezeichnet) für das Laden von Web-Schriftarten. Sie würden den Bucket, der die Web-Schriftart hostet, deshalb so konfigurieren, dass jeder Ursprung diese Anfragen machen kann.

Wie wertet Amazon S3 die CORS-Konfiguration für einen Bucket aus?

Wenn Amazon S3 eine Preflight-Anfrage von einem Browser erhält, wertet es die CORS-Konfiguration für den Bucket aus und verwendet die erste `CORSRule`-Regel, die mit der eingehenden Browser-Anfrage übereinstimmt, um eine ursprungsübergreifende Anfrage zuzulassen. Für die Übereinstimmung mit einer Regel müssen die folgenden Bedingungen erfüllt sei:

- Der `Origin`-Header der Anfrage muss mit einem `AllowedOrigin`-Element übereinstimmen.
- Die Anfragemethode (z. B. GET oder PUT) oder der `Access-Control-Request-Method`-Header bei einer Preflight-`OPTIONS`-Anfrage muss eines der `AllowedMethod`-Elemente sein.

- Jeder im `Access-Control-Request-Headers`-Header der Preflight-Anfrage muss mit einem `AllowedHeader`-Element übereinstimmen.

> **Note**
>
> Alle ACLs und Richtlinien gelten weiterhin, wenn Sie CORS für den Bucket aktivieren.

So unterstützt Object Lambda Access Point CORS

Wenn S3 Object Lambda eine Anforderung von einem Browser empfängt oder die Anforderung einen `Origin`-Header enthält, fügt S3 Object Lambda immer das Header-Feld `“AllowedOrigins":"*"` hinzu.

Weitere Informationen zur Verwendung von CORS finden Sie in den folgenden Themen.

Themen

- CORS-Konfiguration
- Cross-Origin Resource Sharing (CORS) konfigurieren

CORS-Konfiguration

Um Ihren Bucket so zu konfigurieren, dass er ursprungsübergreifende Anfragen zulässt, erstellen Sie eine CORS-Konfiguration. Die CORS-Konfiguration ist ein Dokument mit Regeln, die die Ursprünge, die auf Ihren Bucket zugreifen dürfen, die Vorgänge (HTTP-Methoden), die die einzelnen Ursprünge unterstützen, sowie weitere operationsspezifische Informationen identifizieren. Sie können der Konfiguration bis zu 100 Regeln hinzufügen. Sie können dem Bucket die CORS-Konfiguration als `cors`-Subressource hinzufügen.

Wenn Sie CORS in der S3-Konsole konfigurieren, müssen Sie JSON verwenden, um eine CORS-Konfiguration zu erstellen. Die neue S3-Konsole unterstützt nur JSON CORS-Konfigurationen.

Weitere Informationen über die CORS-Konfiguration und die darin enthaltenen Elemente finden Sie in den folgenden Themen. Anweisungen zum Hinzufügen einer CORS-Konfiguration finden Sie unter Cross-Origin Resource Sharing (CORS) konfigurieren.

> **Important**
>
> In der S3-Konsole muss die CORS-Konfiguration JSON sein.

Themen

Beispiel 1

Statt über einen Amazon-S3-Website-Endpunkt auf eine Website zuzugreifen, können Sie Ihre eigen Domäne verwenden, wie beispielsweise `example1.com`, um Ihren Inhalt bereitzustellen. Weitere Informationen zur Verwendung Ihrer eigenen Domäne finden Sie unter Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne.

Die folgende Beispielkonfiguration für `cors` umfasst drei Regeln, die als `CORSRule`-Elemente angegeben sind:

- Die erste Regel gestattet ursprungsübergreifende PUT-, POST- und DELETE-Anfragen vom Ursprung `http://www.example1.com`. Die Regel gestattet auch alle Header in einer Preflight-OPTIONS-Anfrage durch den `Access-Control-Request-Headers`-Header. Als Antwort auf Preflight-OPTIONS-Anfragen gibt Amazon S3 angeforderte Header zurück.
- Die zweite Regel gestattet dieselben ursprungsübergreifenden Anfragen wie die erste Regel, aber sie bezieht sich auf einen anderen Ursprung, `http://www.example2.com`.
- Die dritte Regel gestattet ursprungsübergreifende GET-Anfragen von allen Ursprüngen. Das Platzhalterzeichen * bezieht sich auf alle Ursprünge.

JSON

```
[
```

```
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST",
            "DELETE"
        ],
        "AllowedOrigins": [
            "http://www.example1.com"
        ],
        "ExposeHeaders": []
    },
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST",
            "DELETE"
        ],
        "AllowedOrigins": [
            "http://www.example2.com"
        ],
        "ExposeHeaders": []
    },
    {
        "AllowedHeaders": [],
        "AllowedMethods": [
            "GET"
        ],
        "AllowedOrigins": [
            "*"
        ],
        "ExposeHeaders": []
    }
]
```

XML

```
<CORSConfiguration>
```

```
 <CORSRule>
   <AllowedOrigin>http://www.example1.com</AllowedOrigin>

   <AllowedMethod>PUT</AllowedMethod>
   <AllowedMethod>POST</AllowedMethod>
   <AllowedMethod>DELETE</AllowedMethod>

   <AllowedHeader>*</AllowedHeader>
 </CORSRule>
 <CORSRule>
   <AllowedOrigin>http://www.example2.com</AllowedOrigin>

   <AllowedMethod>PUT</AllowedMethod>
   <AllowedMethod>POST</AllowedMethod>
   <AllowedMethod>DELETE</AllowedMethod>

   <AllowedHeader>*</AllowedHeader>
 </CORSRule>
 <CORSRule>
   <AllowedOrigin>*</AllowedOrigin>
   <AllowedMethod>GET</AllowedMethod>
 </CORSRule>
</CORSConfiguration>
```

Beispiel 2

Die CORS-Konfiguration unterstützt auch optionale Konfigurationsparameter, wie in der folgenden CORS-Konfiguration gezeigt. In diesem Beispiel gestattet die folgende CORS-Konfiguration ursprungsübergreifende PUT-, POST- und DELETE-Anfragen vom Ursprung `http://www.example.com`.

JSON

```
[
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST",
            "DELETE"
```

```
        ],
        "AllowedOrigins": [
            "http://www.example.com"
        ],
        "ExposeHeaders": [
            "x-amz-server-side-encryption",
            "x-amz-request-id",
            "x-amz-id-2"
        ],
        "MaxAgeSeconds": 3000
    }
]
```

XML

```
<CORSConfiguration>
 <CORSRule>
   <AllowedOrigin>http://www.example.com</AllowedOrigin>
   <AllowedMethod>PUT</AllowedMethod>
   <AllowedMethod>POST</AllowedMethod>
   <AllowedMethod>DELETE</AllowedMethod>
   <AllowedHeader>*</AllowedHeader>
  <MaxAgeSeconds>3000</MaxAgeSeconds>
  <ExposeHeader>x-amz-server-side-encryption</
ExposeHeader>
  <ExposeHeader>x-amz-request-id</
ExposeHeader>
  <ExposeHeader>x-amz-id-2</ExposeHeader>
 </CORSRule>
</CORSConfiguration>
```

Das `CORSRule`-Element in der obigen Konfiguration beinhaltet die folgenden optionalen Elemente:

- `MaxAgeSeconds` – Gibt den Zeitraum in Sekunden an (in diesem Beispiel 3000), für den der Browser eine Amazon-S3-Antwort auf eine Preflight-OPTIONS-Anfrage für die angegebene Ressource zwischenspeichert. Durch die Zwischenspeicherung der Antwort muss der Browser keine Preflight-Anfragen an Amazon S3 senden, wenn die ursprüngliche Anfrage wiederholt werden soll.

- `ExposeHeader`– Identifiziert die Antwort-Header (in diesem Beispiel , `x-amz-server-side-encryption` `x-amz-request-id`und `x-amz-id-2`), auf die Kunden von ihren Anwendungen aus zugreifen können (z. B. von einem JavaScript `XMLHttpRequest` -Objekt).

AllowedMethod -Element

In der CORS-Konfiguration können Sie die folgenden Werte für das `AllowedMethod`-Element angeben.

- GET
- PUT
- POST
- DELETE
- HEAD

AllowedOrigin -Element

Im `AllowedOrigin`-Element geben Sie die Ursprünge an, über die Sie domänenübergreifende Anfragen erlauben möchten, z. B. `http://www.example.com`. Die Ursprungszeichenfolge darf nur ein `*`-Platzhalterzeichen enthalten, wie beispielsweise `http://*.example.com`. Optional können Sie `*` als Ursprung angeben, sodass alle Ursprünge ursprungsübergreifende Anfragen senden dürfen. Sie können auch `https` angeben, um nur sichere Ursprünge zuzulassen.

AllowedHeader -Element

Das `AllowedHeader`-Element gibt an, welche Header in einer Preflight-Anfrage durch den `Access-Control-Request-Headers`-Header erlaubt sind. Jeder Header-Name im `Access-Control-Request-Headers`-Header muss mit einem entsprechenden Eintrag in der Regel übereinstimmen. Amazon S3 sendet nur die zulässigen angeforderten Header in einer Antwort. Eine Liste mit Beispielen für Header, die in Anfragen an Amazon S3 verwendet werden können, finden Sie unter Häufig verwendete Anforderungsheader im API-Referenzhandbuch zum Amazon Simple Storage Service.

Jede AllowedHeader Zeichenfolge in der Regel darf höchstens ein * Platzhalterzeichen enthalten. Beispielsweise aktiviert `<AllowedHeader>x-amz-*</AllowedHeader>` alle für Amazon spezifischen Header.

ExposeHeader -Element

Jedes `ExposeHeader` Element identifiziert einen Header in der Antwort, auf den Kunden von ihren Anwendungen aus zugreifen können sollen (z. B. von einem JavaScript `XMLHttpRequest` Objekt). Eine Liste der gängigen Amazon-S3-Antwortheader finden Sie unter Häufig verwendete Anforderungsheader im API-Referenzhandbuch zum Amazon Simple Storage Service.

MaxAgeSeconds -Element

Das `MaxAgeSeconds`-Element gibt die Zeit in Sekunden an, wie lange Ihr Browser die Antwort auf eine Preflight-Anfrage zwischenspeichern kann, wie nach der Ressource, der HTTP-Methode und dem Ursprung identifiziert.

Cross-Origin Resource Sharing (CORS) konfigurieren

Cross-Origin Resource Sharing (CORS) bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain. Mit CORS-Unterstützung können Sie umfassende clientseitige Webanwendungen mit Amazon S3 erstellen und selektiven ursprungsübergreifenden Zugriff auf Ihre Amazon-S3-Ressourcen zulassen.

In diesem Abschnitt erfahren Sie, wie Sie CORS mithilfe der Amazon S3-Konsole, der Amazon S3-REST-API und der AWS SDKs aktivieren. Um Ihren Bucket so zu konfigurieren, dass er ursprungsübergreifende Anfragen zulässt, fügen Sie dem Bucket eine CORS-Konfiguration hinzu. Eine CORS-Konfiguration ist ein Dokument, das Regeln, die die Ursprünge identifizieren, die den Zugriff auf Ihren Bucket zulassen, die Vorgänge (HTTP-Methoden), die die einzelnen Ursprünge unterstützen, sowie weitere operationsspezifische Informationen definiert. In der S3-Konsole muss die CORS-Konfiguration ein JSON-Dokument sein.

Beispiele für CORS-Konfigurationen in JSON und XML finden Sie unter CORS-Konfiguration.

Verwenden der S3-Konsole

Dieser Abschnitt erklärt, wie Sie die Amazon-S3-Konsole verwenden, um einem S3-Bucket CORS (Cross-Origin Resource Sharing) hinzuzufügen.

Alle Zugriffskontrolllisten (ACLs) und andere Zugriffsberechtigungsrichtlinien gelten weiterhin, wenn Sie CORS für den Bucket aktivieren.

> **Important**
>
> In der neuen S3-Konsole muss die CORS-Konfiguration JSON sein. Beispiele für CORS-Konfigurationen in JSON und XML finden Sie unter CORS-Konfiguration.

So fügen Sie einem S3-Bucket eine CORS-Konfiguration hinzu:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie eine Bucket-Richtlinie erstellen wollen.
3. Wählen Sie Permissions (Berechtigungen).
4. Wählen Sie im Abschnitt Cross-Origin Resource Sharing (CORS) die Option Edit (Bearbeiten) aus.
5. Geben Sie in das Textfeld CORS configuration editor eine neue CORS-Konfiguration ein, fügen Sie dort eine kopierte Konfiguration ein oder bearbeiten Sie eine vorhandene Konfiguration.

   Die CORS-Konfiguration ist eine JSON-Datei. Der Text, den Sie in den Editor eingeben, muss gültiges JSON sein. Weitere Informationen finden Sie unter CORS-Konfiguration.
6. Wählen Sie Save Changes (Änderungen speichern).

> **Note**
>
> Amazon S3 zeigt den Amazon Resource Name (ARN) für den Bucket neben dem Titel CORS configuration editor an. Weitere Informationen zu ARNs finden Sie unter Amazon-Ressourcennamen (ARNs) und AWS Service-Namespaces im Allgemeine Amazon Web Services-Referenz.

Verwenden der AWS SDKs

Sie können das AWS SDK verwenden, um Cross-Origin Resource Sharing (CORS) für einen Bucket zu verwalten. Weitere Informationen über CORS finden Sie unter Cross-Origin Resource Sharing (CORS) verwenden.

Im Folgenden sind einige Beispiele aufgeführt:

- Erstellt eine CORS-Konfiguration und legt die Konfiguration für einen Bucket fest.
- Ruft die Konfiguration ab und ändert sie durch Hinzufügen einer Regel ab
- Fügt die abgeänderte Konfiguration dem Bucket hinzu
- Löscht die Konfiguration

Java

Example

Example

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketCrossOriginConfiguration;
import com.amazonaws.services.s3.model.CORSRule;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class CORS {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        // Create two CORS rules.
        List<CORSRule.AllowedMethods> rule1AM = new
 ArrayList<CORSRule.AllowedMethods>();
        rule1AM.add(CORSRule.AllowedMethods.PUT);
        rule1AM.add(CORSRule.AllowedMethods.POST);
        rule1AM.add(CORSRule.AllowedMethods.DELETE);
```

```
        CORSRule rule1 = new
CORSRule().withId("CORSRule1").withAllowedMethods(rule1AM)
                .withAllowedOrigins(Arrays.asList("http://*.example.com"));

        List<CORSRule.AllowedMethods> rule2AM = new
ArrayList<CORSRule.AllowedMethods>();
        rule2AM.add(CORSRule.AllowedMethods.GET);
        CORSRule rule2 = new
CORSRule().withId("CORSRule2").withAllowedMethods(rule2AM)
                .withAllowedOrigins(Arrays.asList("*")).withMaxAgeSeconds(3000)
                .withExposedHeaders(Arrays.asList("x-amz-server-side-encryption"));

        List<CORSRule> rules = new ArrayList<CORSRule>();
        rules.add(rule1);
        rules.add(rule2);

        // Add the rules to a new CORS configuration.
        BucketCrossOriginConfiguration configuration = new
BucketCrossOriginConfiguration();
        configuration.setRules(rules);

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Add the configuration to the bucket.
            s3Client.setBucketCrossOriginConfiguration(bucketName, configuration);

            // Retrieve and display the configuration.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            printCORSConfiguration(configuration);

            // Add another new rule.
            List<CORSRule.AllowedMethods> rule3AM = new
ArrayList<CORSRule.AllowedMethods>();
            rule3AM.add(CORSRule.AllowedMethods.HEAD);
            CORSRule rule3 = new
CORSRule().withId("CORSRule3").withAllowedMethods(rule3AM)
                    .withAllowedOrigins(Arrays.asList("http://www.example.com"));

            rules = configuration.getRules();
            rules.add(rule3);
```

```
            configuration.setRules(rules);
            s3Client.setBucketCrossOriginConfiguration(bucketName, configuration);

            // Verify that the new rule was added by checking the number of rules in
the
            // configuration.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            System.out.println("Expected # of rules = 3, found " +
configuration.getRules().size());

            // Delete the configuration.
            s3Client.deleteBucketCrossOriginConfiguration(bucketName);
            System.out.println("Removed CORS configuration.");

            // Retrieve and display the configuration to verify that it was
            // successfully deleted.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            printCORSConfiguration(configuration);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void printCORSConfiguration(BucketCrossOriginConfiguration
configuration) {
        if (configuration == null) {
            System.out.println("Configuration is null.");
        } else {
            System.out.println("Configuration has " +
configuration.getRules().size() + " rules\n");

            for (CORSRule rule : configuration.getRules()) {
                System.out.println("Rule ID: " + rule.getId());
                System.out.println("MaxAgeSeconds: " + rule.getMaxAgeSeconds());
                System.out.println("AllowedMethod: " + rule.getAllowedMethods());
                System.out.println("AllowedOrigins: " + rule.getAllowedOrigins());
                System.out.println("AllowedHeaders: " + rule.getAllowedHeaders());
                System.out.println("ExposeHeader: " + rule.getExposedHeaders());
```

```
                System.out.println();
            }
        }
    }
}
```

.NET

Example

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CORSTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CORSConfigTestAsync().Wait();
        }
        private static async Task CORSConfigTestAsync()
        {
            try
            {
                // Create a new configuration request and add two rules
                CORSConfiguration configuration = new CORSConfiguration
                {
                    Rules = new System.Collections.Generic.List<CORSRule>
```

```
                        {
                          new CORSRule
                          {
                            Id = "CORSRule1",
                            AllowedMethods = new List<string> {"PUT", "POST",
 "DELETE"},
                            AllowedOrigins = new List<string> {"http://
*.example.com"}
                          },
                          new CORSRule
                          {
                            Id = "CORSRule2",
                            AllowedMethods = new List<string> {"GET"},
                            AllowedOrigins = new List<string> {"*"},
                            MaxAgeSeconds = 3000,
                            ExposeHeaders = new List<string> {"x-amz-server-side-
encryption"}
                          }
                        }
                };

                // Add the configuration to the bucket.
                await PutCORSConfigurationAsync(configuration);

                // Retrieve an existing configuration.
                configuration = await RetrieveCORSConfigurationAsync();

                // Add a new rule.
                configuration.Rules.Add(new CORSRule
                {
                    Id = "CORSRule3",
                    AllowedMethods = new List<string> { "HEAD" },
                    AllowedOrigins = new List<string> { "http://www.example.com" }
                });

                // Add the configuration to the bucket.
                await PutCORSConfigurationAsync(configuration);

                // Verify that there are now three rules.
                configuration = await RetrieveCORSConfigurationAsync();
                Console.WriteLine();
                Console.WriteLine("Expected # of rulest=3; found:{0}",
 configuration.Rules.Count);
                Console.WriteLine();
```

```
                Console.WriteLine("Pause before configuration delete. To continue,
click Enter...");
                Console.ReadKey();

                // Delete the configuration.
                await DeleteCORSConfigurationAsync();

                // Retrieve a nonexistent configuration.
                configuration = await RetrieveCORSConfigurationAsync();
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        static async Task PutCORSConfigurationAsync(CORSConfiguration configuration)
        {

            PutCORSConfigurationRequest request = new PutCORSConfigurationRequest
            {
                BucketName = bucketName,
                Configuration = configuration
            };

            var response = await s3Client.PutCORSConfigurationAsync(request);
        }

        static async Task<CORSConfiguration> RetrieveCORSConfigurationAsync()
        {
            GetCORSConfigurationRequest request = new GetCORSConfigurationRequest
            {
                BucketName = bucketName

            };
            var response = await s3Client.GetCORSConfigurationAsync(request);
            var configuration = response.Configuration;
            PrintCORSRules(configuration);
```

```
            return configuration;
        }

        static async Task DeleteCORSConfigurationAsync()
        {
            DeleteCORSConfigurationRequest request = new
 DeleteCORSConfigurationRequest
            {
                BucketName = bucketName
            };
            await s3Client.DeleteCORSConfigurationAsync(request);
        }

        static void PrintCORSRules(CORSConfiguration configuration)
        {
            Console.WriteLine();

            if (configuration == null)
            {
                Console.WriteLine("\nConfiguration is null");
                return;
            }

            Console.WriteLine("Configuration has {0} rules:",
 configuration.Rules.Count);
            foreach (CORSRule rule in configuration.Rules)
            {
                Console.WriteLine("Rule ID: {0}", rule.Id);
                Console.WriteLine("MaxAgeSeconds: {0}", rule.MaxAgeSeconds);
                Console.WriteLine("AllowedMethod: {0}", string.Join(", ",
 rule.AllowedMethods.ToArray()));
                Console.WriteLine("AllowedOrigins: {0}", string.Join(", ",
 rule.AllowedOrigins.ToArray()));
                Console.WriteLine("AllowedHeaders: {0}", string.Join(", ",
 rule.AllowedHeaders.ToArray()));
                Console.WriteLine("ExposeHeader: {0}", string.Join(", ",
 rule.ExposeHeaders.ToArray()));
            }
        }
    }
}
```

Verwenden der REST-API

Sie können die AWS Management Console verwenden, um eine CORS-Konfiguration für Ihren Bucket zu erstellen. Falls in Ihrer Anwendung erforderlich, können Sie auch direkt REST-Anfragen senden. In den folgenden Abschnitten der API-Referenz zum Amazon Simple Storage Service werden die REST-API-Aktionen im Zusammenhang mit der CORS-Konfiguration beschrieben:

- PutBucketCors
- GetBucketCors
- DeleteBucketCors
- OPTIONS-Objekt

Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher

Die Amazon S3 Block Public Access-Funktion bietet Einstellungen für Zugriffspunkte, Buckets und Konten, mit denen Sie den öffentlichen Zugriff auf Amazon-S3-Ressourcen verwalten können. Standardmäßig erlauben neue Buckets, Zugriffspunkte und Objekte keinen öffentlichen Zugriff. Benutzer können jedoch Bucket-Richtlinien, Zugriffspunkt-Richtlinien oder Objektberechtigungen ändern, um öffentlichen Zugriff zu ermöglichen. S3 Block Public Access-Einstellungen überschreiben diese Richtlinien und Berechtigungen, damit Sie den öffentlichen Zugriff auf diese Ressourcen einschränken können.

Mit S3 Block Public Access können Administratoren und Bucket-Eigentümer problemlos zentrale Kontrollen zur Begrenzung des öffentlichen Zugriffs auf ihre Amazon-S3-Ressourcen einrichten. Diese Kontrollen werden unabhängig davon durchgesetzt, wie die Ressourcen erstellt wurden.

Anweisungen zur Konfiguration von Public Block Access finden Sie unter Verwenden von Block Public Access.

Wenn Amazon S3 eine Anforderung zum Zugriff auf einen Bucket oder ein Objekt erhält, wird ermittelt, ob für den Bucket oder das Konto des Bucket-Eigentümers eine Block Public Access-Einstellung vorliegt. Wenn die Anforderung über einen Zugriffspunkt einging, prüft Amazon S3 auch auf Block Public Access-Einstellungen für den Zugriffspunkt. Wenn eine Block Public Access-Einstellung vorhanden ist, die den angeforderten Zugriff verbietet, lehnt Amazon S3 die Anforderung ab.

Amazon S3 Block Public Access bietet vier Einstellungen. Diese Einstellungen sind voneinander unabhängig und können in beliebiger Kombination verwendet werden. Jede Einstellung kann auf

einen Zugriffspunkt, einen Bucket oder ein gesamtes AWS-Konto angewendet werden. Wenn sich die Block Public Access-Einstellungen für den Zugriffspunkt, den Bucket oder das Konto unterscheiden, wendet Amazon S3 die restriktivste Kombination der Zugriffspunkt-, Bucket- und Kontoeinstellungen an.

Wenn Amazon S3 ermittelt, ob eine Operation von einer Block Public Access-Einstellung untersagt wird, werden alle Anforderungen abgelehnt, die gegen eine Zugriffspunkt-, Bucket- oder Konto-Einstellung verstoßen.

Warning

Öffentlicher Zugriff für Buckets und Objekte wird über Zugriffskontrolllisten (ACLs), Zugriffspunktrichtlinien und Bucket-Richtlinien gewährt. Um sicherzustellen, dass der öffentliche Zugriff für alle Amazon-S3-Zugriffspunkte, -Buckets und -Objekte blockiert ist, empfehlen wir, alle vier Einstellungen zur Blockierung des öffentlichen Zugriffs für das Konto zu aktivieren. Diese Einstellungen blockieren den öffentlichen Zugriff für alle aktuellen und künftigen Buckets und Zugriffspunkte.
Bevor Sie diese Einstellungen anwenden, verifizieren Sie, dass Ihre Anwendungen ohne öffentlichen Zugriff korrekt funktionieren. Wenn ein bestimmter Umfang an öffentlichem Zugriff auf Ihre Buckets oder Objekte nötig ist, z. B. zum Hosten einer statischen Website, wie unter Hosten einer statischen Website mit Amazon S3 beschrieben, können Sie die einzelnen Einstellungen an Ihre Speicheranwendungsfälle anpassen.

Note

- Sie können die Einstellungen für Block Public Access nur für Zugangspunkte, Buckets und AWS-Konten aktivieren. Amazon S3 unterstützt keine Block Public Access-Einstellungen für den öffentlichen Zugriff pro Objekt.
- Wenn Sie Block Public Access-Einstellungen auf ein Konto anwenden, gelten die Einstellungen AWS-Regionen global für alle . Die Einstellungen werden möglicherweise nicht in allen Regionen umgehend oder gleichzeitig wirksam, werden aber auf jeden Fall von allen Regionen übernommen.

Themen

- Block Public Access-Einstellungen

- Durchführen von Block Public Access-Vorgänge an einem Zugriffspunkt
- Die Bedeutung von „öffentlich“
- Verwenden von IAM Access Analyzer für S3 zur Überprüfung öffentlicher Buckets
- Berechtigungen
- Verwenden von Block Public Access
- Konfigurieren der Block-Public-Access-Einstellungen für Ihr Konto
- Konfigurieren von Block-Public-Access-Einstellungen für Ihre S3-Buckets

Block Public Access-Einstellungen

S3 Block Public Access bietet vier Einstellungen. Sie können diese Einstellungen in beliebiger Kombination auf einzelne Zugriffspunkte, Buckets oder auf ganze AWS-Konten anwenden. Wenn Sie eine Einstellung auf ein Konto anwenden, gilt sie für alle Buckets und Zugriffspunkte, die dem Konto gehören. Wenn Sie eine Einstellung auf einen Bucket anwenden, gilt diese auch für alle Zugriffspunkte, die diesem Bucket zugeordnet sind.

Die folgende Tabelle enthält die verfügbaren Einstellungen.

Name	Beschreibung
`BlockPublicAcls`	Das Festlegen dieser Option auf `TRUE` hat folgende Verhaltensweise zur Folge: • "PUT Bucket acl"- und "PUT Object acl"-Aufrufe schlagen fehl, wenn die angegebene Access Control List (ACL) öffentlich ist. • "PUT Object"-Aufrufe schlagen fehl, wenn die Anforderung eine öffentliche ACL enthält. • Wenn diese Einstellung auf ein Konto angewendet wird, schlagen "PUT Bucket"-Aufrufe fehl, wenn die Anforderung eine öffentliche ACL enthält. Wenn diese Einstellung auf festgelegt ist`TRUE`, schlagen die angegebenen Operationen fehl (gleich AWS CLI, AWS ob sie über die REST-API oder SDKs erfolgen). Vorhandene Richtlinien und ACLs für Buckets und

Name	Beschreibung
	Objekte werden jedoch nicht geändert. Mit dieser Einstellung können Sie den öffentlichen Zugriff einschränken und gleichzeitig die vorhandenen Richtlinien und ACLs für Ihre Buckets und Objekte prüfen, verbessern oder anderweitig ändern. **Note** Zugriffspunkten sind keine ACLs zugeordnet. Wenn Sie diese Einstellung auf einen Zugriffspunkt anwenden, fungiert sie als Passthrough zum zugrunde liegenden Bucket. Wenn diese Einstellung für einen Zugriffspunkt aktiviert ist, verhalten sich Anforderungen, die über den Zugriffspunkt vorgenommen werden, so als ob der zugrunde liegende Bucket diese Einstellung aktiviert hat, unabhängig davon, ob diese Einstellung für den Bucket tatsächlich aktiviert ist.
`IgnorePublicAcls`	Wenn Sie diese Option auf `TRUE` festlegen, ignoriert Amazon S3 alle öffentlichen ACLs auf einem Bucket und alle darin enthaltenen Objekte. Mit dieser Einstellung können Sie den von ACLs gewährten öffentlichen Zugriff sicher blockieren und gleichzeitig "PUT Object"-Aufrufe zulassen, die eine öffentliche ACL enthalten (im Gegensatz zu `BlockPublicAcls` , das "PUT Object"-Aufrufe zurückweist, die eine öffentliche ACL enthalten). Die Aktivierung dieser Einstellungen wirkt sich nicht auf die Persistenz von vorhandenen ACLs aus und verhindert nicht, dass neue öffentliche ACLs eingerichtet werden. **Note** Zugriffspunkten sind keine ACLs zugeordnet. Wenn Sie diese Einstellung auf einen Zugriffspunkt anwenden, fungiert sie als Passthrough zum zugrunde liegenden Bucket. Wenn diese Einstellung für einen Zugriffspunkt aktiviert ist, verhalten sich Anforderungen, die über den Zugriffspunkt vorgenommen werden, so als ob der zugrunde liegende Bucket diese Einstellung aktiviert hat, unabhängig davon, ob diese Einstellung für den Bucket tatsächlich aktiviert ist.

Name	Beschreibung
`BlockPublicPolicy`	Wenn Sie diese Option für einen Bucket auf `TRUE` festlegen, lehnt Amazon S3 Aufrufe der PUT-Bucket-Richtlinie ab, wenn die angegebene Bucket-Richtlinie öffentlichen Zugriff zulässt. Wenn Sie diese Option für einen Bucket auf `TRUE` einstellen, lehnt Amazon S3 Aufrufe der PUT-Zugriffspunktrichtlinie für alle Zugriffspunkte des Buckets im selben Konto ab, wenn die angegebene Richtlinie öffentlichen Zugriff zulässt. Wenn Sie diese Option für einen Zugriffspunkt auf `TRUE` festlegen, lehnt Amazon S3 Aufrufe der PUT-Zugriffspunkt-Richtlinie und der PUT-Bucket-Richtlinie ab, die über den Zugriffspunkt vorgenommen werden, wenn die angegebene Richtlinie (für den Zugriffspunkt oder den zugrunde liegenden Bucket) öffentlichen Zugriff erlaubt. Mit dieser Einstellung können Sie Benutzern erlauben, Zugriffspunkt- und Bucket-Richtlinien zu verwalten, ohne ihnen die öffentliche Freigabe des Buckets oder der darin enthaltenen Objekte zu gestatten. Die Aktivierung dieser Einstellung hat keine Auswirkungen auf vorhandene Zugriffspunkt- oder Bucket-Richtlinien. **Important** Um diese Einstellung effektiv zu nutzen, empfehlen wir Ihnen, sie auf Konto-Ebene anzuwenden. Eine Bucket-Richtlinie kann Benutzern das Ändern der Block Public Access-Einstellungen eines Buckets gestatten. Daher könnten Benutzer mit der Berechtigung zum Ändern einer Bucket-Richtlinie eine Richtlinie einfügen, die es ihnen erlaubt, die Block Public Access-Einstellungen für den Bucket zu deaktivieren. Wenn diese Einstellung dann für das ganze Konto aktiviert wird statt für einen bestimmten Bucket, blockiert Amazon S3 öffentliche Richtlinien selbst dann, wenn ein Benutzer die Bucket-Richtlinie so ändert, dass diese Einstellung deaktiviert wird.

Name	Beschreibung
`RestrictPublicBuckets`	Wenn Sie diese Option auf festlegen, wird der Zugriff auf einen Zugriffspunkt oder Bucket mit einer öffentlichen Richtlinie nur auf AWS Service-Prinzipale und autorisierte Benutzer innerhalb des Kontos des Bucket-Eigentümers und des Kontos des Zugriffspunktbesitzers `TRUE` beschränkt. Diese Einstellung blockiert den gesamten kontoübergreifenden Zugriff auf den Zugriffspunkt oder Bucket (außer durch AWS Service-Prinzipale), während Benutzer innerhalb des Kontos weiterhin den Zugriffspunkt oder Bucket verwalten können. Die Aktivierung dieser Einstellung wirkt sich nicht auf vorhandene Zugriffspunkt- oder Bucket-Richtlinien aus – mit folgender Ausnahme: Amazon S3 blockiert den öffentlichen und kontenübergreifenden Zugriff, der von einer öffentlichen Zugriffspunkt- oder Bucket-Richtlinie abgeleitet wird, darunter auch die nicht-öffentliche Delegation auf bestimmte Konten.

Important

- "GET Bucket acl"- und "GET Object acl"-Aufrufe geben immer die effektiven Berechtigungen für den angegebenen Bucket oder das angegebene Objekt zurück. Nehmen wir als Beispiel an, dass ein Bucket über eine ACL verfügt, die den öffentlichen Zugriff zulässt, dass für den Bucket aber auch die Einstellung `IgnorePublicAcls` aktiviert ist. In diesem Fall gibt „GET Bucket acl“ statt der ACL, die tatsächlich mit dem Bucket verknüpft ist, eine ACL zurück, die die von Amazon S3 durchgesetzten Zugriffsberechtigungen widerspiegelt.
- Die Block Public Access-Einstellungen ändern keine vorhandenen Richtlinien oder ACLs. Daher sorgt das Entfernen einer Block Public Access-Einstellung dafür, dass ein Bucket oder ein Objekt mit einer öffentlichen Richtlinie oder ACLs wieder öffentlich zugänglich ist.

Durchführen von Block Public Access-Vorgänge an einem Zugriffspunkt

Um Block Public Access-Operationen auf einem Zugriffspunkt durchzuführen, verwenden Sie den AWS CLI Service `s3control`.

> **Important**
>
> Beachten Sie, dass es derzeit nicht möglich ist, die Block Public Accesss-Einstellungen eines Zugriffspunkts nach dem Erstellen des Zugriffspunkts zu ändern. Die einzige Möglichkeit, Block Public Access-Einstellungen für einen Zugriffspunkt anzugeben, besteht darin, diese beim Erstellen des Zugriffspunkts einzuschließen.

Die Bedeutung von „öffentlich“

ACLs

Amazon S3 betrachtet eine Bucket- oder Objekt-ACL als öffentlich, wenn sie Mitgliedern der vordefinierten Gruppen `AllUsers` oder `AuthenticatedUsers` irgendwelche Berechtigungen erteilt. Weitere Informationen zu vordefinierten Gruppen finden Sie unter Vordefinierte Gruppen in Amazon S3.

Bucket-Richtlinien

Bei der Evaluierung einer Bucket-Richtlinie beginnt Amazon S3 mit der Annahme, dass die Richtlinie öffentlich ist. Dann evaluiert es die Richtlinie, um festzustellen, ob sie als nicht-öffentlich eingestuft werden kann. Um als nicht öffentlich zu gelten, darf eine Bucket-Richtlinie nur Zugriff auf feste Werte (Werte, die keine Platzhalter oder eine AWS Identity and Access Management -Richtlinienvariable enthalten) gewähren, für einen oder mehrere der folgenden Werte:

- Ein AWS -Prinzipal, Benutzer, Rolle oder Service-Prinzipal (z. B. `aws:PrincipalOrgID`)
- einen Satz von Classless Inter-Domain Routings (CIDRs), unter Verwendung von `aws:SourceIp`. Weitere Informationen zu CIDR finden Sie unter RFC 4632 auf der RFC-Editor-Website.

  > **Note**
  >
  > Bucket-Richtlinien, die abhängig vom `aws:SourceIp`-Bedingungsschlüssel Zugriff mit sehr breiten IP-Bereichen gewähren (z. B. 0.0.0.0/1), werden als „öffentlich“ bewertet. Dazu gehören Werte, die breiter sind als /8 für IPv4 und /32 für IPv6 (ausgenommen private RFC1918-Bereiche). Die Funktion zum Blockieren des öffentlichen Zugriffs lehnt diese „öffentlichen“ Richtlinien ab und verhindert den kontoübergreifenden Zugriff auf Buckets, die diese „öffentlichen“ Richtlinien bereits verwenden.

- `aws:SourceArn`

- `aws:SourceVpc`
- `aws:SourceVpce`
- `aws:SourceOwner`
- `aws:SourceAccount`
- `s3:x-amz-server-side-encryption-aws-kms-key-id`
- `aws:userid`, außerhalb des Musters `"AROLEID:*"`
- `s3:DataAccessPointArn`

> **Note**
>
> Bei Verwendung in einer Bucket-Richtlinie kann dieser Wert einen Platzhalter für den Namen des Zugriffspunkts enthalten, ohne die Richtlinie öffentlich zu machen, solange die Konto-ID fixiert ist. Das Zulassen des Zugriffs auf `arn:aws:s3:us-west-2:123456789012:accesspoint/*` würde beispielsweise den Zugriff auf jeden Zugriffspunkt ermöglichen, der dem Konto `123456789012` in Region `us-west-2` zugeordnet ist, ohne die Bucket-Richtlinie öffentlich zu machen. Beachten Sie, dass sich dieses Verhalten von dem von Zugriffspunkt-Richtlinien unterscheidet. Weitere Informationen finden Sie unter Zugriffspunkte.

- `s3:DataAccessPointAccount`

Weitere Informationen zu Bucket-Richtlinien finden Sie unter Bucket-Richtlinien und Benutzerrichtlinien.

Example : Öffentliche Bucket-Richtlinien

Nach diesen Regeln gelten die folgenden Beispielrichtlinien als öffentlich.

```
{
  "Principal": { "Federated": "graph.facebook.com" },
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow"
 }
```

```
{
```

```
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow"
 }
```

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow",
  "Condition": { "StringLike": {"aws:SourceVpc": "vpc-*"}}
 }
```

Diese Richtlinien können zu nicht-öffentlichen Richtlinien gemacht werden, indem einer der vorgenannten Bedingungsschlüssel unter Verwendung eines festen Wertes eingefügt wird. So können Sie beispielsweise die letzte oben angeführte Richtlinie zu einer nicht-öffentlichen Richtlinie machen, indem Sie `aws:SourceVpc` wie folgt auf einen festen Wert festlegen.

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow",
  "Condition": {"StringEquals": {"aws:SourceVpc": "vpc-91237329"}}
 }
```

So wertet Amazon S3 eine Bucket-Richtlinie aus, die Berechtigungen sowohl für den öffentlichen wie auch für den nicht-öffentlichen Zugriff enthält

Dieses Beispiel zeigt, wie Amazon S3 eine Bucket-Richtlinie bewertet, die Berechtigungen sowohl für den öffentlichen wie auch für den nicht-öffentlichen Zugriff enthält.

Nehmen wir an, dass ein Bucket eine Richtlinie aufweist, die den Zugriff auf einen Satz an festen Prinzipalen gewährt. Nach den zuvor beschriebenen Regeln gilt diese Richtlinie nicht als öffentlich. Wenn Sie daher die Einstellung `RestrictPublicBuckets` aktivieren, bleibt die Richtlinie wie geschrieben in Kraft, da `RestrictPublicBuckets` nur für Buckets mit öffentlichen Richtlinien gilt. Wenn Sie jedoch der Richtlinie eine öffentliche Anweisung hinzufügen, wird `RestrictPublicBuckets` für den Bucket wirksam. Es erlaubt nur AWS Service-Prinzipalen und autorisierten Benutzern des Kontos des Bucket-Eigentümers, auf den Bucket zuzugreifen.

Nehmen wir beispielsweise an, dass ein Bucket, der „Konto-1“ gehört, eine Richtlinie aufweist, die Folgendes enthält:

1. Eine Anweisung, die Zugriff auf gewährt AWS CloudTrail (bei dem es sich um einen - AWS Service-Prinzipal handelt)
2. eine Anweisung, die Konto "Konto-2" Zugriff gewährt
3. eine Anweisung, die der Öffentlichkeit Zugriff gewährt, z. B. durch das Festlegen von `"Principal": "*"` ohne einschränkende `Condition`

Diese Richtlinie gilt wegen der dritten Anweisung als öffentlich. Wenn diese Richtlinie vorhanden und `RestrictPublicBuckets` aktiviert ist, erlaubt Amazon S3 den Zugriff nur durch CloudTrail. Beachten Sie: Obwohl die zweite Anweisung nicht öffentlich ist, deaktiviert Amazon S3 den Zugriff durch „Account-2“. Das liegt daran, dass die dritte Anweisung die gesamte Richtlinie zu einer öffentlichen Richtlinie macht, so dass `RestrictPublicBuckets` gilt. Somit deaktiviert Amazon S3 den kontoübergreifenden Zugriff, obwohl die Richtlinie den Zugriff an ein bestimmtes Konto („Account-2“) delegiert. Wenn Sie aber die dritte Anweisung aus der Richtlinie entfernen, gilt die Richtlinie nicht als öffentlich, und `RestrictPublicBuckets` ist nicht mehr gültig. Somit erhält „Konto-2“ wieder Zugriff auf den Bucket, selbst wenn Sie `RestrictPublicBuckets` aktiviert lassen.

Zugriffspunkte

Amazon S3 wertet Block Public Access-Einstellungen für Zugriffspunkte geringfügig anders als für Buckets aus. Die Regeln, die Amazon S3 anwendet, um zu bestimmen, wann eine Zugriffspunkt-Richtlinie öffentlich ist, sind für Zugriffspunkte im Allgemeinen dieselben wie für Buckets, außer in den folgenden Situationen:

- Ein Zugriffspunkt mit einem VPC-Netzwerkursprung wird unabhängig vom Inhalt seiner Zugriffspunkt-Richtlinie immer als nicht öffentlich betrachtet.
- Eine Zugriffspunktrichtlinie, die Zugriff auf eine Gruppe von Zugriffspunkten unter Verwendung von `s3:DataAccessPointArn` gewährt, gilt als öffentlich. Beachten Sie, dass sich dieses Verhalten von dem von Bucket-Richtlinien unterscheidet. Beispielsweise wird eine Bucket-Richtlinie, die Zugriff auf `s3:DataAccessPointArn`-Werte gewährt, die `arn:aws:s3:us-west-2:123456789012:accesspoint/*` entsprechen, nicht als öffentlich betrachtet. Dieselbe Anweisung in einer Zugriffspunkt-Richtlinie würde jedoch den Zugriffspunkt öffentlich machen.

Verwenden von IAM Access Analyzer für S3 zur Überprüfung öffentlicher Buckets

Sie können IAM Access Analyzer für S3 verwenden, um Buckets mit Bucket-ACLs, Bucket-Richtlinien oder Zugriffspunktrichtlinien zu überprüfen, die öffentlichen Zugriff gewähren. IAM Access Analyzer für S3 macht Sie auf Buckets aufmerksam, die so konfiguriert sind, dass jedem im Internet oder anderen , einschließlich AWS-Konten außerhalb Ihrer Organisation AWS-Konten, Zugriff gewährt wird. Für jeden öffentlichen oder freigegebenen Bucket erhalten Sie Ergebnisse, die die Quelle und die Ebene des öffentlichen oder freigegebenen Zugriffs melden.

In IAM Access Analyzer für S3 können Sie den gesamten öffentlichen Zugriff auf einen Bucket mit einem einzigen Klick blockieren. Sie können auch einen Drilldown in die Berechtigungseinstellungen auf Bucket-Ebene ausführen, um detaillierte Zugriffsebenen zu konfigurieren. Für bestimmte und geprüfte Anwendungsfälle, die öffentlichen oder freigegebenen Zugriff erfordern, können Sie Ihre Absicht bestätigen und aufzeichnen, dass der Bucket öffentlich oder freigegeben bleibt, indem Sie die Ergebnisse für den Bucket archivieren.

In seltenen Fällen meldet IAM Access Analyzer für S3 möglicherweise keine Erkenntnisse für einen Bucket, den eine Bewertung von Amazon S3 Block Public Access als öffentlich meldet. Dies ist der Fall, da Amazon S3 Block Public Access Richtlinien für aktuelle Aktionen und potenzielle Aktionen, die in Zukunft hinzugefügt werden könnten und die dazu führen könnten, dass ein Bucket öffentlich wird, überprüft. Andererseits analysiert IAM Access Analyzer für S3 nur die aktuellen Aktionen, die für den Amazon-S3-Service bei der Bewertung des Zugriffsstatus festgelegt wurden.

Weitere Informationen zu IAM Access Analyzer für S3 finden Sie unter Überprüfen des Bucket-Zugriffs mit IAM Access Analyzer für S3.

Berechtigungen

Um die Funktionen von Amazon S3 Block Public Access zu nutzen, benötigen Sie die folgenden Berechtigungen:

Operation	Erforderliche Berechtigungen
GET bucket policy status	`s3:GetBucketPolicyStatus`
GET bucket Block Public Access settings	`s3:GetBucketPublicAccessBlock`
PUT bucket Block Public Access settings	`s3:PutBucketPublicAccessBlock`
DELETE bucket Block Public Access settings	`s3:PutBucketPublicAccessBlock`

Operation	Erforderliche Berechtigungen
GET account Block Public Access settings	`s3:GetAccountPublicAccessBlock`
PUT account Block Public Access settings	`s3:PutAccountPublicAccessBlock`
DELETE account Block Public Access settings	`s3:PutAccountPublicAccessBlock`
PUT Zugriffspunkt-Block Public Access-Einstellungen	`s3:CreateAccessPoint`

> **Note**
>
> Für die DELETE-Vorgänge sind dieselben Berechtigungen erforderlich wie für die PUT-Vorgänge. Es gibt keine separaten Berechtigungen für die DELETE-Vorgänge.

Verwenden von Block Public Access

Weitere Informationen zum Konfigurieren von Block Public Access für Ihr AWS-Konto und Ihre Amazon S3-Buckets finden Sie in den folgenden Themen.

- Konfigurieren der Block-Public-Access-Einstellungen für Ihr Konto
- Konfigurieren von Block-Public-Access-Einstellungen für Ihre S3-Buckets

Konfigurieren der Block-Public-Access-Einstellungen für Ihr Konto

Die Amazon-S3-Block-Public-Access-Funktion bietet Einstellungen für Zugriffspunkte, Buckets und Konten, mit denen Sie den öffentlichen Zugriff auf Amazon-S3-Ressourcen verwalten können. Standardmäßig erlauben neue Buckets, Zugriffspunkte und Objekte keinen öffentlichen Zugriff.

Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

> **Note**
>
> Einstellungen auf Kontoebene haben Vorrang vor Einstellungen für einzelne Objekte. Wenn Sie Ihr Konto so konfigurieren, dass der öffentliche Zugriff gesperrt wird, werden

alle Einstellungen für den öffentlichen Zugriff, die für einzelne Objekte in Ihrem Konto vorgenommen wurden, außer Kraft gesetzt.

Sie können die S3-Konsole, AWS CLI, AWS SDKs und die REST-API verwenden, um Einstellungen für das Blockieren des öffentlichen Zugriffs für alle Buckets in Ihrem Konto zu konfigurieren. Weitere Informationen dazu finden Sie in den folgenden Abschnitten.

Informationen zum Konfigurieren von Block Public Access-Einstellungen für Ihre Buckets finden Sie unter Konfigurieren von Block-Public-Access-Einstellungen für Ihre S3-Buckets. Weitere Hinweise zu Zugriffspunkten finden Sie unter Durchführen von Block Public Access-Vorgänge an einem Zugriffspunkt.

Verwenden der S3-Konsole

Amazon S3 Block Public Access verhindert die Anwendung von Einstellungen, die den öffentlichen Zugriff auf Daten in S3-Buckets erlauben. In diesem Abschnitt wird erläutert, wie Sie die Block-Public-Access-Einstellungen für alle S3-Buckets in Ihrem AWS-Konto bearbeiten. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

So bearbeiten Sie die Block Public Access-Einstellungen für alle S3-Buckets in einem AWS-Konto

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie Block Public Access settings for this account (Block Public Access-Einstellung für dieses Konto).
3. Wählen Sie Edit (Bearbeiten) aus, um die Block-Public-Access-Einstellungen für alle Buckets in Ihrem AWS-Konto zu ändern.
4. Wählen Sie die Einstellung aus, die Sie ändern möchten, und wählen Sie anschließend Save changes(Änderungen speichern) aus.
5. Wenn Sie zur Bestätigung aufgefordert werden, geben Sie **`confirm`** ein. Wählen Sie anschließend Confirm (Bestätigen) aus, um Ihre Änderungen zu speichern.

Verwenden der AWS CLI

Sie können Amazon S3 Block Public Access über die AWS CLI verwenden. Weitere Informationen zum Einrichten und Verwenden der AWS CLI finden Sie unter Was ist die AWS Command Line Interface?

Account

Um Block Public Access-Vorgänge auf einem Konto durchzuführen, verwenden Sie den AWS CLI - Service `s3control`. Folgende Vorgänge auf Konto-Ebene verwenden diesen Service:

- PUT PublicAccessBlock (für ein Konto)
- GET PublicAccessBlock (für ein Konto)
- DELETE PublicAccessBlock (für ein Konto)

Weitere Informationen und Beispiele finden Sie unter put-public-access-block in der AWS CLI - Referenz.

Verwenden der AWS SDKs

Java

Die folgenden Beispiele zeigen Ihnen, wie Sie Amazon S3 Block Public Access mit verwenden AWS SDK for Java , um eine Blockkonfiguration für den öffentlichen Zugriff auf ein Amazon S3-Konto zu setzen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwendung der AWS SDK for Java.

```
AWSS3ControlClientBuilder controlClientBuilder =
 AWSS3ControlClientBuilder.standard();
controlClientBuilder.setRegion(<region>);
controlClientBuilder.setCredentials(<credentials>);

AWSS3Control client = controlClientBuilder.build();
client.putPublicAccessBlock(new PutPublicAccessBlockRequest()
  .withAccountId(<account-id>)
  .withPublicAccessBlockConfiguration(new PublicAccessBlockConfiguration()
    .withIgnorePublicAcls(<value>)
    .withBlockPublicAcls(<value>)
    .withBlockPublicPolicy(<value>)
    .withRestrictPublicBuckets(<value>)));
```

> **Important**
>
> Dieses Beispiel trifft nur für Vorgänge auf Konto-Ebene zu, die die `AWSS3Control`-Client-Klasse verwenden. Sehen Sie sich für Vorgänge auf Bucket-Ebene das vorhergehende Beispiel an.

Other SDKs

Informationen zur Verwendung der anderen AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Verwenden der REST-API

Informationen zur Verwendung von Amazon S3 Block Public Access über die REST-APIs finden Sie in den folgenden Themen der Amazon Simple Storage Service API-Referenz.

- Vorgänge auf Konto-Ebene
  - PUT PublicAccessBlock
  - GET PublicAccessBlock
  - LÖSCHEN PublicAccessBlock

Konfigurieren von Block-Public-Access-Einstellungen für Ihre S3-Buckets

Die Amazon-S3-Block-Public-Access-Funktion bietet Einstellungen für Zugriffspunkte, Buckets und Konten, mit denen Sie den öffentlichen Zugriff auf Amazon-S3-Ressourcen verwalten können. Standardmäßig erlauben neue Buckets, Zugriffspunkte und Objekte keinen öffentlichen Zugriff.

Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Sie können die S3-Konsole, AWS CLI AWS SDKs und die REST-API verwenden, um öffentlichen Zugriff auf einen oder mehrere Buckets zu gewähren. Sie können den öffentlichen Zugriff auf Buckets, die bereits öffentlich sind, auch blockieren. Weitere Informationen dazu finden Sie in den folgenden Abschnitten.

Informationen zum Konfigurieren von Einstellungen zum Blockieren des öffentlichen Zugriffs für jeden Bucket in Ihrem Konto finden Sie unter Konfigurieren der Block-Public-Access-Einstellungen für Ihr Konto. Informationen zum Konfigurieren von Block Public Access für Zugriffspunkte finden Sie unter Durchführen von Block Public Access-Vorgänge an einem Zugriffspunkt.

Verwenden der S3-Konsole

Amazon S3 Block Public Access verhindert die Anwendung von Einstellungen, die den öffentlichen Zugriff auf Daten in S3-Buckets erlauben. In diesem Abschnitt wird erläutert, wie Sie die Block-Public-Access-Einstellungen für einen oder mehrere S3-Buckets bearbeiten. Informationen zum Blockieren des öffentlichen Zugriffs mithilfe der AWS CLI, AWS SDKs und der Amazon S3-REST-APIs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

In der Liste der Buckets können Sie sehen, ob Ihr Bucket öffentlich zugänglich ist. In der Spalte Zugriff kennzeichnet Amazon S3 die Berechtigungen für einen Bucket wie folgt:

- Public (Öffentlich) – Alle Benutzer haben Zugriff auf eines oder mehrere der folgenden Elemente: Objekte auflisten, Objekte schreiben, Lese- und Schreibberechtigungen.
- Objects can be public (Objekte können öffentlich sein) – Der Bucket ist nicht öffentlich, aber jeder, der über die entsprechenden Berechtigungen verfügt, kann den öffentlichen Zugriff auf Objekte gewähren.
- Buckets and objects not public (Buckets und Objekte sind nicht öffentlich) – Für den Bucket und die Objekte gilt kein öffentlicher Zugriff.
- Nur autorisierte Benutzer dieses Kontos – Der Zugriff ist auf IAM-Benutzer und -Rollen in diesem Konto und diesen AWS Service-Prinzipalen beschränkt, da es eine Richtlinie gibt, die öffentlichen Zugriff gewährt.

Sie können Bucket-Suchen auch nach Zugriffstyp filtern. Wählen Sie aus der Dropdown-Liste neben der Leiste Search for buckets (Nach Buckets suchen) einen Zugriffstyp aus.

Wenn beim Auflisten Ihrer Buckets mit ihren Einstellungen für den öffentlichen Zugriff die Fehlermeldung `Error` angezeigt wird, verfügen Sie möglicherweise nicht über die erforderlichen Berechtigungen. Stellen Sie sicher, dass Sie Ihrer Benutzer- oder Rollenrichtlinie die folgenden Berechtigungen hinzugefügt haben:

```
s3:GetAccountPublicAccessBlock
s3:GetBucketPublicAccessBlock
s3:GetBucketPolicyStatus
```

```
s3:GetBucketLocation
s3:GetBucketAcl
s3:ListAccessPoints
s3:ListAllMyBuckets
```

In einigen seltenen Fällen können Anfragen auch aufgrund des Ausfalls einer AWS-Region fehlschlagen.

Bearbeiten der Amazon-S3-Block-Public-Access-Einstellungen für einen einzelnen S3-Bucket

Führen Sie die folgenden Schritte aus, wenn Sie die Einstellungen für den öffentlichen Zugriff für einen einzelnen S3-Bucket ändern müssen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket-Name den Namen des gewünschten Buckets aus.
3. Wählen Sie Permissions (Berechtigungen).
4. Wählen Sie Edit (Bearbeiten) aus, um die Einstellungen für den öffentlichen Zugriff für den Bucket zu ändern. Weitere Informationen zu den vier Amazon-S3-Block-Public-Access-Einstellungen finden Sie unter Block Public Access-Einstellungen.
5. Wählen Sie die Einstellung aus, die Sie ändern möchten, und wählen Sie anschließend Save (Speichern) aus.
6. Wenn Sie zur Bestätigung aufgefordert werden, geben Sie **`confirm`** ein. Wählen Sie anschließend Confirm (Bestätigen) aus, um Ihre Änderungen zu speichern.

Sie können beim Erstellen eines Buckets die Amazon S3 Block Public Access-Einstellungen ändern. Weitere Informationen finden Sie unter Erstellen eines Buckets.

Verwenden der AWS CLI

Um den öffentlichen Zugriff auf einen Bucket zu blockieren oder den Block für den öffentlichen Zugriff zu löschen, verwenden Sie den AWS CLI Service `s3api`. Folgende Vorgänge auf Bucket-Ebene verwenden diesen Service:

- PUT PublicAccessBlock (für einen Bucket)
- GET PublicAccessBlock (für einen Bucket)
- DELETE PublicAccessBlock (für einen Bucket)
- GET BucketPolicyStatus

Weitere Informationen und Beispiele finden Sie unter put-public-access-block in der AWS CLI - Referenz.

Verwenden der AWS SDKs

Java

```
AmazonS3 client = AmazonS3ClientBuilder.standard()
   .withCredentials(<credentials>)
   .build();

client.setPublicAccessBlock(new SetPublicAccessBlockRequest()
  .withBucketName(<bucket-name>)
  .withPublicAccessBlockConfiguration(new PublicAccessBlockConfiguration()
    .withBlockPublicAcls(<value>)
    .withIgnorePublicAcls(<value>)
    .withBlockPublicPolicy(<value>)
    .withRestrictPublicBuckets(<value>)));
```

Important

Dieses Beispiel trifft nur für Vorgänge auf Bucket-Ebene zu, die die `AmazonS3`-Client-Klasse verwenden. Sehen Sie sich für Vorgänge auf Konto-Ebene das folgende Beispiel an.

Other SDKs

Informationen zur Verwendung der anderen AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Verwenden der REST-API

Informationen zur Verwendung von Amazon S3 Block Public Access über die REST-APIs finden Sie in den folgenden Themen der Amazon Simple Storage Service API-Referenz.

- Vorgänge auf Bucket-Ebene
  - PUT PublicAccessBlock

- GET PublicAccessBlock
- DELETE PublicAccessBlock
- GET BucketPolicyStatus

Überprüfen des Bucket-Zugriffs mit IAM Access Analyzer für S3

IAM Access Analyzer für S3 macht Sie auf S3-Buckets aufmerksam, die so konfiguriert sind, dass jedem im Internet oder anderen , einschließlich AWS-Konten außerhalb Ihrer Organisation AWS-Konten, Zugriff gewährt wird. Für jeden öffentlichen oder freigegebenen Bucket erhalten Sie Ergebnisse bezüglich der Quelle und der Ebene des öffentlichen oder freigegebenen Zugriffs. Beispielsweise könnte IAM Access Analyzer für S3 zeigen, dass ein Bucket über Lese- oder Schreibzugriff verfügt, der über eine Bucket-Zugriffssteuerungsliste (ACL), eine Bucket-Richtlinie, eine Richtlinie für Multi-Region Access Points oder eine Zugriffspunktrichtlinie bereitgestellt wird. Mit diesem Erkenntnissen können Sie sofortige und präzise Korrekturmaßnahmen ergreifen, um den Bucket-Zugriff wie beabsichtigt wiederherzustellen.

Wenn Sie einen gefährdeten Bucket in IAM Access Analyzer für S3 überprüfen, können Sie den gesamten öffentlichen Zugriff auf den Bucket mit einem einzigen Klick blockieren. Wir empfehlen Ihnen, den gesamten Zugriff auf Ihre Buckets zu blockieren, es sei denn, Sie benötigen öffentlichen Zugriff, um einen bestimmten Anwendungsfall zu unterstützen. Bevor Sie den gesamten öffentlichen Zugriff blockieren, stellen Sie sicher, dass Ihre Anwendungen ohne öffentlichen Zugriff weiterhin ordnungsgemäß funktionieren. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Sie können auch einen Drilldown in die Berechtigungseinstellungen auf Bucket-Ebene ausführen, um detaillierte Zugriffsebenen zu konfigurieren. Für bestimmte und überprüfte Anwendungsfälle, die öffentlichen Zugriff erfordern, wie statisches Website-Hosting, öffentliche Downloads oder kontenübergreifende Freigabe, können Sie Ihre Absicht bestätigen und aufzeichnen, dass der Bucket öffentlich oder freigegeben bleibt, indem Sie die Ergebnisse für den Bucket archivieren. Diese Bucket-Konfigurationen sind jederzeit aufrufbar und änderbar. Sie können Ihre Ergebnisse auch als CSV-Bericht zu Auditing-Zwecken herunterladen.

IAM Access Analyzer für S3 ist ohne zusätzliche Kosten in der Amazon-S3-Konsole verfügbar. IAM Access Analyzer für S3 wird von AWS Identity and Access Management (IAM) IAM Access Analyzer bereitgestellt. Um IAM Access Analyzer für S3 in der Amazon S3-Konsole zu verwenden, müssen Sie die IAM-Konsole aufrufen und IAM Access Analyzer pro Region aktivieren.

Weitere Informationen zu IAM Access Analyzer finden Sie unter Was ist IAM Access Analyzer? im IAM-Benutzerhandbuch. Weitere Informationen zu IAM Access Analyzer für S3 finden Sie in den folgenden Abschnitten.

Important

- IAM Access Analyzer für S3 erfordert einen Analyzer auf Kontoebene. Um IAM Access Analyzer für S3 zu verwenden, müssen Sie IAM Access Analyzer aufrufen und einen Analysator erstellen, der ein Konto als Vertrauenszone hat. Weitere Informationen finden Sie unter Aktivieren von IAM Access Analyzer im IAM-Benutzerhandbuch.
- IAM Access Analyzer für S3 analysiert nicht die Zugriffspunktrichtlinie, die kontoübergreifenden Zugriffspunkten angefügt ist. Dieses Verhalten tritt auf, weil sich der Zugriffspunkt und seine Richtlinie außerhalb der Vertrauenszone, d. h. des Kontos, befinden. Buckets, die den Zugriff an einen kontoübergreifenden Zugriffspunkt delegieren, werden unter Buckets mit öffentlichem Zugriff aufgeführt, sofern Sie die Einstellung `RestrictPublicBuckets` zum Blockieren des öffentlichen Zugriffs nicht auf den Bucket oder das Konto angewendet haben. Wenn Sie die Einstellung „Öffentlichen Zugriff `RestrictPublicBuckets` blockieren“ anwenden, wird der Bucket unter Buckets mit Zugriff von anderen AWS-Konten – einschließlich von Drittanbietern – AWS-Kontengemeldet.
- Wenn eine Bucket-Richtlinie oder Bucket-ACL hinzugefügt oder geändert wird, generiert und aktualisiert IAM Access Analyzer Erkenntnisse basierend auf der Änderung innerhalb von 30 Minuten. Ergebnisse im Zusammenhang mit den Einstellungen zum Blockieren des öffentlichen Zugriffs auf Kontoebene werden möglicherweise erst bis zu 6 Stunden, nachdem Sie die Einstellungen geändert haben, generiert oder aktualisiert. Ergebnisse, die sich auf Multi-Regions-Zugriffspunkte beziehen, können bis zu sechs Stunden nach dem Erstellen, Löschen oder Ändern der Richtlinie des Multi-Region Access Points nicht generiert oder aktualisiert werden.

Topics

- Archivieren von Bucket-Ergebnissen
- Aktivieren eines archivierten Bucket-Ergebnisses
- Anzeigen von Ergebnisdetails
- Herunterladen eines Berichts von IAM Access Analyzer für S3

Welche Informationen stellt IAM Access Analyzer für S3 zur Verfügung?

IAM Access Analyzer für S3 liefert Erkenntnisse für Buckets, auf die außerhalb Ihres AWS-Konto s zugegriffen werden kann. Buckets, die unter Buckets with public access (Buckets mit öffentlichem Zugriff) aufgeführt sind, können jedem im Internet zugänglich. Wenn IAM Access Analyzer für S3 öffentliche Buckets identifiziert, wird oben auf der Seite eine Warnung angezeigt, die Ihnen die Anzahl der öffentlichen Buckets in Ihrer Region anzeigt. Buckets, die unter Buckets mit Zugriff von anderen AWS-Konten – einschließlich Drittanbietern AWS-Konten aufgeführt sind, werden bedingt mit anderen geteilt AWS-Konten, einschließlich Konten außerhalb Ihrer Organisation.

Für jeden Bucket bietet IAM Access Analyzer für S3 die folgenden Informationen:

- Bucket-Name
- Von Access Analyzer entdeckt – Gibt an, wann IAM Access Analyzer für S3 den öffentlichen oder gemeinsamen Bucket-Zugriff entdeckt hat.
- Shared through (Freigabe über) – Wie der Bucket freigegeben wird – über eine Bucket-Richtlinie, eine Bucket-ACL, eine Multi-Regions-Zugriffspunkt-Richtlinie oder eine Zugriffspunkt-Richtlinie. Multi-Region Access Points und kontoübergreifende Zugriffspunkte werden unter Zugriffspunkten aufgeführt. Ein Bucket kann sowohl über Richtlinien als auch über ACLs freigegeben werden. Wenn Sie die Quelle für den Bucket-Zugriff suchen und überprüfen möchten, können die Informationen in dieser Spalte als Ausgangspunkt für sofortige und präzise Korrekturmaßnahmen dienen.
- Status - Der Status der Erkenntnisse zu dem Bucket. IAM Access Analyzer für S3 zeigt Erkenntnisse für alle öffentlichen und gemeinsam genutzten Buckets an.
  - Active (Aktiv) – Ergebnis wurde nicht überprüft.
  - Archived (Archiviert) – Das Ergebnis wurde wie vorgesehen überprüft und bestätigt.
  - All e– Alle Ergebnisse für Buckets, die öffentlich oder für andere freigegeben sind AWS-Konten, einschließlich AWS-Konten außerhalb Ihrer Organisation.
- Access level (Zugriffsebene) – Gewährte Zugriffsberechtigungen für den Bucket:
  - List (Liste) – Auflistung der Ressourcen.

- Read (Lesen) – Lesen aber kein Bearbeiten von Ressourceninhalten und -attributen.
- Write (Schreiben) – Erstellen, Löschen oder Ändern von Ressourcen.
- Permissions (Berechtigungen) – Erteilen oder ändern der Ressourcenberechtigungen.
- Markierungen – Aktualisieren der einer Ressource zugeordneten Markierungen.

Aktivieren von IAM Access Analyzer für S3

Um IAM Access Analyzer für S3 zu verwenden, müssen Sie die folgenden erforderlichen Schritte ausführen.

1. Sie müssen die erforderlichen Berechtigungen erteilen.

   Weitere Informationen finden Sie unter Zur Verwendung von IAM Access Analyzer erforderliche Berechtigungen im IAM-Benutzerhandbuch.
2. Rufen Sie IAM auf, um einen Analyzer auf Kontoebene für jede Region zu erstellen, in der Sie IAM Access Analyzer verwenden möchten.

   IAM Access Analyzer für S3 erfordert einen Analyzer auf Kontoebene. Um IAM Access Analyzer für S3 verwenden zu können, müssen Sie einen Analyzer erstellen, der ein Konto als Vertrauenszone hat. Weitere Informationen finden Sie unter Aktivieren von IAM Access Analyzer im IAM-Benutzerhandbuch.

Blockieren des gesamten öffentlichen Zugriffs

Wenn Sie den gesamten Zugriff auf einen Bucket mit einem einzigen Klick blockieren möchten, können Sie die Schaltfläche Blockieren des gesamten öffentlichen Zugriffs in IAM Access Analyzer für S3 verwenden. Wenn Sie den gesamten öffentlichen Zugriff auf einen Bucket blockieren, wird kein öffentlicher Zugriff gewährt. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren, es sei denn, Sie benötigen öffentlichen Zugriff, um einen bestimmten und verifizierten Anwendungsfall zu unterstützen. Bevor Sie den gesamten öffentlichen Zugriff blockieren, stellen Sie sicher, dass Ihre Anwendungen ohne öffentlichen Zugriff weiterhin ordnungsgemäß funktionieren.

Wenn Sie nicht den gesamten öffentlichen Zugriff auf Ihren Bucket blockieren möchten, können Sie Ihre Block Public Access-Einstellungen in der Amazon-S3-Konsole bearbeiten, um detaillierte Zugriffsebenen für Ihre Buckets zu konfigurieren. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

In seltenen Fällen meldet IAM Access Analyzer für S3 möglicherweise keine Erkenntnisse für einen Bucket, den eine Bewertung von Amazon S3 Block Public Access als öffentlich meldet. Dies ist der Fall, da Amazon S3 Block Public Access Richtlinien für aktuelle Aktionen und potenzielle Aktionen, die in Zukunft hinzugefügt werden könnten und die dazu führen könnten, dass ein Bucket öffentlich wird, überprüft. Andererseits analysiert IAM Access Analyzer für S3 nur die aktuellen Aktionen, die für den Amazon-S3-Service bei der Bewertung des Zugriffsstatus festgelegt wurden.

So blockieren Sie den gesamten öffentlichen Zugriff auf einen Bucket mit IAM Access Analyzer für S3

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich auf der linken Seite unter Dashboards die Option Access Analyzer for S3 aus.
3. Wählen Sie in IAM Access Analyzer für S3 einen Bucket aus.
4. Wählen Sie Block all public access (Öffentlichen Zugriff blockieren) aus.
5. Um zu bestätigen, dass Sie den gesamten öffentlichen Zugriff auf den Bucket blockieren möchten, geben Sie unter Block all public access (bucket settings) (Öffentlichen Zugriff blockieren) (Bucket-Einstellungen) **`confirm`** ein.

   Amazon S3 blockiert den gesamten öffentlichen Zugriff auf Ihren Bucket. Der Status der Bucket-Erkenntnis wird auf gelöst aktualisiert und der Bucket erscheint nicht mehr in der Liste von IAM Access Analyzer für S3. Wenn Sie gelöste Buckets überprüfen möchten, öffnen Sie IAM Access Analyzer in der IAM-Konsole.

Überprüfen und Ändern des Bucket-Zugriffs

Wenn Sie nicht beabsichtigt haben, öffentlichen oder anderen AWS-Konten, einschließlich Konten außerhalb Ihrer Organisation, Zugriff zu gewähren, können Sie die Bucket-ACL, die Bucket-Richtlinie, die Richtlinie für Multi-Region Access Points oder die Zugriffspunktrichtlinie ändern, um den Zugriff auf den Bucket zu entfernen. In der Spalte Shared through (Freigegeben durch) werden alle Quellen des Bucket-Zugriffs angezeigt: Bucket-Richtlinie, Bucket-ACL und/oder Zugriffspunkt-Richtlinie. Multi-Region Access Points und kontoübergreifende Zugriffspunkte werden unter Zugriffspunkten aufgeführt.

So überprüfen und ändern Sie eine Bucket-Richtlinie, eine Bucket-Zugriffskontrollliste, ein Multi-Regions-Zugriffspunkt oder eine Zugriffspunkt-Richtlinie

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Access Analyzer for S3.
3. Um zu sehen, ob öffentlicher Zugriff oder freigegebener Zugriff über eine Bucket-Richtlinie, eine Bucket-ACL, eine Multi-Regions-Zugriffspunkt-Richtlinie oder eine Zugriffspunkt-Richtlinie gewährt wird, nehmen Sie auf die Spalte Shared through (Freigegeben durch) Bezug.
4. Wählen Sie unter Buckets den Namen des Buckets mit der Bucket-Richtlinie, der Bucket-Zugriffskontrollliste, der Multi-Regions-Zugriffspunkt-Richtlinie oder der Zugriffspunkt-Richtlinie aus, die Sie ändern oder überprüfen möchten.
5. Wenn Sie eine Bucket-ACL ändern oder anzeigen möchten:

   a. Wählen Sie Permissions (Berechtigungen).

   b. Wählen Sie Access Control List.

   c. Überprüfen Sie Ihre Bucket-ACL und nehmen Sie bei Bedarf Änderungen vor.

      Weitere Informationen finden Sie unter Konfigurieren von ACLs.
6. Wenn Sie eine Bucket-Richtlinie ändern oder überprüfen möchten, gehen Sie folgendermaßen vor:

   a. Wählen Sie Permissions (Berechtigungen).

   b. Wählen Sie Bucket Policy aus.

   c. Überprüfen oder ändern Sie Ihre Bucket-Richtlinie nach Bedarf.

      Weitere Informationen finden Sie unter Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole.
7. Wenn Sie eine Multi-Regions-Zugriffspunkt-Richtlinie ändern oder anzeigen, gehen Sie folgendermaßen vor:

   a. Wählen Sie Multiregionaler Zugriffspunkt.

   b. Wählen Sie den multiregionalen Zugriffspunktsnamen.

   c. Überprüfen oder ändern Sie die Multi-Regions-Zugriffspunkt-Richtlinie nach Bedarf.

      Weitere Informationen finden Sie unter Berechtigungen.

8. Wenn Sie eine Zugriffspunkt-Richtlinie überprüfen oder ändern möchten, gehen Sie folgendermaßen vor:

   a. Wählen Sie Access Points (Zugriffspunkte).

   b. Wählen Sie den Namen des Zugriffspunkts aus.

   c. Überprüfen oder ändern Sie den Zugriff nach Bedarf.

      Weitere Informationen finden Sie unter Nutzen von Amazon S3-Zugriffspunkten mit der Amazon S3-Konsole.

   Wenn Sie eine Bucket-ACL, eine Bucket-Richtlinie oder eine Zugriffspunkt-Richtlinie bearbeiten oder entfernen, um den öffentlichen oder gemeinsamen Zugriff zu entfernen, wird der Status für die Bucket-Ergebnisse auf gelöst aktualisiert. Die aufgelösten Bucket-Ergebnisse verschwinden aus der Liste von IAM Access Analyzer für S3, aber Sie können sie in IAM Access Analyzer anzeigen.

Archivieren von Bucket-Ergebnissen

Wenn ein Bucket Zugriff auf öffentliche oder andere AWS-Konten, einschließlich Konten außerhalb Ihrer Organisation, gewährt, um einen bestimmten Anwendungsfall zu unterstützen (z. B. eine statische Website, öffentliche Downloads oder kontoübergreifende Freigabe), können Sie das Ergebnis für den Bucket archivieren. Wenn Sie Bucket-Ergebnisse archivieren, bestätigen und verzeichnen Sie Ihre Absicht, dass der Bucket öffentlich oder freigegeben bleiben soll. Archivierte Bucket-Erkenntnisse verbleiben in Ihrer Liste von IAM Access Analyzer für S3, sodass Sie immer wissen, welche Buckets öffentlich oder freigegeben sind.

So archivieren Sie Bucket-Erkenntnisse in IAM Access Analyzer für S3

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Access Analyzer for S3.
3. Wählen Sie in IAM Access Analyzer für S3 einen aktiven Bucket aus.
4. Um Ihre Absicht zu bestätigen, dass auf diesen Bucket von der öffentlichen oder anderen AWS-Konten, einschließlich Konten außerhalb Ihrer Organisation, zugegriffen werden kann, wählen Sie Archivieren.
5. Geben Sie **`confirm`** ein und wählen Sie Archiv (Archivieren).

Aktivieren eines archivierten Bucket-Ergebnisses

Nachdem Sie Ergebnisse archiviert haben, können Sie sie jederzeit erneut einsehen und ihren Status wieder auf aktiv ändern, wodurch angegeben wird, dass für den Bucket eine weitere Überprüfung erforderlich ist.

So aktivieren Sie eine archivierte Bucket-Erkenntnis in IAM Access Analyzer für S3

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Access Analyzer for S3.
3. Wählen Sie die archivierten Bucket-Ergebnisse aus.
4. Wählen Sie Mark as active (Als aktiv markieren) aus.

Anzeigen von Ergebnisdetails

Wenn Sie weitere Informationen zu einem Bucket benötigen, können Sie die Details der Bucket-Erkenntnis in IAM Access Analyzer in der IAM-Konsole öffnen.

So zeigen Sie Erkenntnisdetails in IAM Access Analyzer für S3 an

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Access Analyzer for S3.
3. Wählen Sie in IAM Access Analyzer für S3 einen Bucket aus.
4. Wählen Sie die Option View details aus.

   Die Erkenntnisdetails werden in IAM Access Analyzer auf der IAM-Konsole geöffnet.

Herunterladen eines Berichts von IAM Access Analyzer für S3

Sie können Ihre Bucket-Ergebnisse als CSV-Bericht herunterladen, den Sie für Auditing-Zwecke verwenden können. Der Bericht enthält die gleichen Informationen, die Sie in IAM Access Analyzer für S3 in der Amazon-S3-Konsole sehen.

So laden Sie einen Bericht herunter

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Access Analyzer for S3 aus.

3. Wählen Sie im Filter „Region“ die Region aus.

   IAM Access Analyzer für S3 wird mit Buckets für die ausgewählte Region aktualisiert.

4. Wählen Sie Download Report (Bericht herunterladen) aus.

   Ein CSV-Bericht wird generiert und auf Ihrem Computer gespeichert.

Überprüfen der Bucket-Eigentümerschaft mit Bucket-Eigentümer-Bedingung

Amazon S3-Bucket-Eigentümerbedingung stellt sicher, dass die Buckets, die Sie in Ihren S3-Operationen verwenden AWS-Konten , zu den von Ihnen erwarteten gehören.

Die meisten S3-Vorgänge lesen oder schreiben in bestimmte S3-Buckets. Zu diesen Vorgängen gehören das Hochladen, Kopieren und Herunterladen von Objekten, das Abrufen oder Ändern von Bucket-Konfigurationen und das Abrufen oder Ändern von Objektkonfigurationen. Wenn Sie diese Vorgänge ausführen, geben Sie den Bucket an, den Sie verwenden möchten, indem Sie seinen Namen in die Anforderung einbeziehen. Um beispielsweise ein Objekt aus S3 abzurufen, stellen Sie eine Anforderung, die den Namen eines Buckets und den Objektschlüssel angibt, der aus diesem Bucket abgerufen werden soll.

Da Amazon S3 Buckets anhand ihrer Namen identifiziert, könnte eine Anwendung, die einen falschen Bucket-Namen in einer Anforderung verwendet, versehentlich Vorgänge gegen einen anderen Bucket als erwartet ausführen. Um unbeabsichtigte Bucket-Interaktionen in solchen Situationen zu vermeiden, können Sie die Bucket-Eigentümer-Bedingung verwenden. Die Bucket-Eigentümer-Bedingung ermöglicht es Ihnen, zu überprüfen, ob der Ziel-Bucket im Besitz des erwarteten AWS-Konto ist, und bietet eine zusätzliche Gewissheit dafür, dass Ihre S3-Vorgänge die von Ihnen beabsichtigten Auswirkungen haben.

Themen

Wann die Bucket-Eigentümer-Bedingung verwendet werden sollte

Wir empfehlen, die Bucket-Eigentümer-Bedingung zu verwenden, wenn Sie einen unterstützten S3-Vorgang ausführen und die Konto-ID des erwarteten Bucket-Eigentümers kennen. Die Bucket-Eigentümer-Bedingung ist für alle S3-Objekt-Vorgänge und die meisten S3-Bucketvorgänge verfügbar. Eine Liste der S3-Vorgänge, die die Bucket-Eigentümerbedingung nicht unterstützen, finden Sie unter Beschränkungen und Einschränkungen.

Um den Vorteil der Verwendung der Bucket-Eigentümerbedingung zu sehen, sollten Sie das folgende Szenario berücksichtigen, an dem AWS Kunden Bea beteiligt ist:

1. Bea entwickelt eine Anwendung, die Amazon S3 verwendet. Während der Entwicklung verwendet Bea ihre Testumgebung, AWS-Konto um einen Bucket mit dem Namen zu erstellen`bea-data-test`, und konfiguriert ihre Anwendung so, dass Anfragen an gestellt werden`bea-data-test`.
2. Bea stellt ihre Anwendung bereit, vergisst jedoch, die Anwendung neu zu konfigurieren, um einen Bucket in ihrem Produktions- AWS-Konto zu verwenden.
3. In der Produktion stellt Beas Anwendung Anforderungen an `bea-data-test`, die erfolgreich sind. Dies führt dazu, dass Produktionsdaten in den Bucket im Testkonto von Bea geschrieben werden.

Bea kann sich vor solchen Situationen schützen, indem sie die Bucket-Eigentümer-Bedingung verwendet. Mit der Bucket-Eigentümerbedingung kann Bea die AWS-Konto ID des erwarteten Bucket-Eigentümers in ihre Anforderungen aufnehmen. Amazon S3 überprüft dann die Konto-ID des Bucket-Eigentümers, bevor es eine Anforderung bearbeitet. Wenn der tatsächliche Bucket-Eigentümer nicht mit dem erwarteten Bucket-Eigentümer übereinstimmt, schlägt die Anforderung fehl.

Wenn Bea die Bucket-Eigentümer-Bedingung verwendet, führt das zuvor beschriebene Szenario nicht dazu, dass die Anwendung von Bea versehentlich in einen Test-Bucket schreibt. Stattdessen schlagen die Anforderungen, die ihre Anwendung in Schritt 3 stellt, mit einer Fehlermeldung `Access Denied` fehl. Durch die Verwendung der Bucket-Eigentümer-Bedingung trägt Bea dazu bei, das Risiko einer versehentlichen Interaktion mit Buckets im falschen AWS-Konto zu eliminieren.

Verifizieren eines Bucket-Eigentümers

Um die Bucket-Eigentümer-Bedingung zu verwenden, schließen Sie in ihre Anforderung einen Parameter ein, der den erwarteten Bucket-Eigentümer angibt. Die meisten S3-Vorgänge betreffen nur einen einzelnen Bucket und erfordern nur diesen einzelnen Parameter, um die Bucket-Eigentümer-Bedingung zu verwenden. Für `CopyObject`-Vorgänge gibt dieser erste Parameter den erwarteten

Eigentümer des Ziel-Buckets an, und Sie schließen einen zweiten Parameter ein, um den erwarteten Eigentümer des Quell-Buckets anzugeben.

Wenn Sie eine Anforderung stellen, die einen Bucket-Eigentümer-Bedingungsparameter enthält, überprüft S3 vor der Verarbeitung der Anforderung die Konto-ID des Bucket-Eigentümers mit dem angegebenen Parameter. Wenn der Parameter mit der Konto-ID des Bucket-Eigentümers übereinstimmt, verarbeitet S3 die Anforderung. Wenn der Parameter nicht mit der Konto-ID des Bucket-Eigentümers übereinstimmt, schlägt die Anforderung mit einer Fehlermeldung `Access Denied` fehl.

Sie können die Bucket-Eigentümerbedingung mit den AWS Command Line Interface (AWS CLI), AWS SDKs und Amazon S3-REST-APIs verwenden. Wenn Sie die Bucket-Eigentümerbedingung mit der AWS CLI und den Amazon S3-REST-APIs verwenden, verwenden Sie die folgenden Parameternamen.

Zugriffsmethode	Parameter für Nicht-Kopiervorgänge	Quellparameter für Kopiervorgang	Zielparameter für Kopiervorgang
AWS CLI	`--expected-bucket-owner`	`--expected-source-bucket-owner`	`--expected-bucket-owner`
Amazon-S3-REST-APIs	Header `x-amz-expected-bucket-owner`	Header `x-amz-source-expected-bucket-owner`	Header `x-amz-expected-bucket-owner`

Die Parameternamen, die erforderlich sind, um die Bucket-Eigentümerbedingung mit den AWS -SDKs zu verwenden, variieren je nach Sprache. Informationen zum Ermitteln der erforderlichen Parameter finden Sie in der SDK-Dokumentation für Ihre gewünschte Sprache. Die SDK-Dokumentation finden Sie unter Erstellungstools auf AWS.

Beispiele

Die folgenden Beispiele zeigen, wie Sie die Bucket-Eigentümerbedingung in Amazon S3 mithilfe der AWS CLI oder der implementieren können AWS SDK for Java 2.x.

Example

Beispiel: Ein Objekt hochladen

Im folgenden Beispiel wird ein Objekt unter Verwendung der Bucket-Eigentümer-Bedingung in S3-Bucket *DOC-EXAMPLE-BUCKET1* hochgeladen, um sicherzustellen, dass sich *DOC-EXAMPLE-BUCKET1* im Besitz des AWS-Konto 111122223333 befindet

AWS CLI

```
aws s3api put-object \
                --bucket DOC-EXAMPLE-BUCKET1 --key exampleobject --
body example_file.txt \
                --expected-bucket-owner 111122223333
```

AWS SDK for Java 2.x

```
public void putObjectExample() {
    S3Client s3Client = S3Client.create();;
    PutObjectRequest request = PutObjectRequest.builder()
            .bucket("DOC-EXAMPLE-BUCKET1")
            .key("exampleobject")
            .expectedBucketOwner("111122223333")
            .build();
    Path path = Paths.get("example_file.txt");
    s3Client.putObject(request, path);
}
```

Example

Beispiel: Kopieren eines Objekts

Im folgenden Beispiel wird das Objekt `object1` vom S3-Bucket *DOC-EXAMPLE-BUCKET1* in den S3-Bucket *DOC-EXAMPLE-BUCKET2* kopiert. Dabei wird die Bucket-Eigentümer-Bedingung verwendet, um sicherzustellen, dass die Buckets den erwarteten Konten gemäß der folgenden Tabelle gehören.

Bucket	Erwarteter Eigentümer
DOC-EXAMPLE-BUCKET1	111122223333
DOC-EXAMPLE-BUCKET2	444455556666

AWS CLI

```
aws s3api copy-object --copy-source DOC-EXAMPLE-BUCKET1/object1 \
                            --bucket DOC-EXAMPLE-BUCKET2 --key object1copy \
                            --expected-source-bucket-owner 111122223333 --expected-
bucket-owner 444455556666
```

AWS SDK for Java 2.x

```
public void copyObjectExample() {
        S3Client s3Client = S3Client.create();
        CopyObjectRequest request = CopyObjectRequest.builder()
                .copySource("DOC-EXAMPLE-BUCKET1/object1")
                .destinationBucket("DOC-EXAMPLE-BUCKET2")
                .destinationKey("object1copy")
                .expectedSourceBucketOwner("111122223333")
                .expectedBucketOwner("444455556666")
                .build();
        s3Client.copyObject(request);
    }
```

Example

Beispiel: Abrufen einer Bucket-Richtlinie

Im folgenden Beispiel wird die Zugriffsrichtlinie für S3-Bucket *DOC-EXAMPLE-BUCKET1* abgerufen, wobei die Bucket-Eigentümerbedingung verwendet wird, um sicherzustellen, dass *DOC-EXAMPLE-BUCKET1* dem AWS-Konto 111122223333 gehört.

AWS CLI

```
aws s3api get-bucket-policy --bucket DOC-EXAMPLE-BUCKET1 --expected-bucket-
owner 111122223333
```

AWS SDK for Java 2.x

```
public void getBucketPolicyExample() {
    S3Client s3Client = S3Client.create();
    GetBucketPolicyRequest request = GetBucketPolicyRequest.builder()
            .bucket("DOC-EXAMPLE-BUCKET1")
            .expectedBucketOwner("111122223333")
```

```
                .build();
        try {
            GetBucketPolicyResponse response = s3Client.getBucketPolicy(request);
        }
        catch (S3Exception e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
    }
```

Beschränkungen und Einschränkungen

Die Amazon-S3-Bucket-Eigentümer-Bedingung hat die folgenden Einschränkungen und Grenzen:

- Der Wert des Bucket-Eigentümer-Bedingungsparameters muss eine AWS-Konto ID (12-stelliger numerischer Wert) sein. Service-Prinzipale werden nicht unterstützt.
- Die Bucket-Eigentümerbedingung ist für CreateBucketListBuckets, oder eine der in AWS S3 Control enthaltenen Operationen nicht verfügbar. Amazon S3 ignoriert alle Bucket-Eigentümer-Bedingungsparameter, die in Anfragen für diese Vorgänge enthalten sind
- Die Bucket-Eigentümer-Bedingung überprüft nur, dass das im Verifizierungsparameter angegebene Konto den Bucket besitzt. Die Bucket-Eigentümer-Bedingung überprüft nicht die Konfiguration des Buckets. Sie garantiert auch nicht, dass die Konfiguration des Buckets bestimmte Bedingungen erfüllt oder mit einem früheren Zustand übereinstimmt.

Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie die Eigentümerschaft von Objekten steuern können, die in Ihre Buckets hochgeladen werden, und Zugriffssteuerungslisten (ACLs) deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff

für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf jedes Objekt in Ihrem Bucket besser steuern, unabhängig davon, wer die Objekte in Ihren Bucket hochgeladen hat.

Object Ownership verfügt über drei Einstellungen, mit denen Sie die Eigentümerschaft von Objekten, die in Ihren Bucket hochgeladen werden, steuern und ACLs deaktivieren oder aktivieren können:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Berechtigungen für Daten im S3-Bucket. Der Bucket verwendet Richtlinien, um die Zugriffssteuerung zu definieren.

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.
- Objekt-Writer – Das AWS-Konto , das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

Für die meisten modernen Anwendungsfälle in S3 empfehlen wir Ihnen, ACLs deaktiviert zu lassen, indem Sie die Einstellung „Bucket-Eigentümer erzwungen“ übernehmen und Ihre Bucket-Richtlinie verwenden, um bei Bedarf Daten mit Benutzern außerhalb Ihres Kontos zu teilen. Dieser Ansatz vereinfacht die Berechtigungsverwaltung. Sie können ACLs sowohl für neu erstellte als auch für bereits vorhandene Buckets deaktivieren. Für neu erstellte Buckets sind ACLs standardmäßig deaktiviert. Im Falle eines vorhandenen Buckets, der bereits Objekte enthält, sind die Objekt- und Bucket-ACLs nach dem Deaktivieren von ACLs nicht mehr Teil einer Zugriffsauswertung, und der Zugriff wird auf der Grundlage von Richtlinien gewährt oder verweigert. Für vorhandene Buckets können Sie ACLs jederzeit wieder aktivieren, nachdem Sie sie deaktiviert haben, und Ihre bereits vorhandenen Bucket- und Objekt-ACLs werden wiederhergestellt.

Bevor Sie ACLs deaktivieren, empfehlen wir Ihnen, Ihre Bucket-Richtlinie zu überprüfen, um sicherzustellen, dass sie alle Möglichkeiten abdeckt, wie Sie außerhalb Ihres Kontos Zugriff auf Ihren Bucket gewähren möchten. Nachdem Sie ACLs deaktiviert haben, akzeptiert Ihr Bucket nur PUT-Anforderungen, die keine ACL angeben, oder PUT-Anforderungen mit ACLs für den Bucket-Eigentümer mit voller Kontrolle, wie z. B. die vordefinierte `bucket-owner-full-control`-ACL oder gleichwertige Formen dieser ACL in XML ausgedrückt. Vorhandene

Anwendungen, die ACLs mit voller Kontrolle des Bucket-Eigentümers unterstützen, haben keine Auswirkungen. -PUTAnforderungen, die andere ACLs enthalten (z. B. benutzerdefinierte Erteilungen für bestimmte AWS-Konten), schlagen fehl und geben einen `400` Fehler mit dem Fehlercode zurück`AccessControlListNotSupported`.

Im Gegensatz dazu akzeptiert und berücksichtigt ein Bucket mit der Einstellung „Bevorzugter Bucket-Eigentümer“ weiterhin Bucket- und Objekt-ACLs. Mit dieser Einstellung gehören neue Objekte, die mit der von `bucket-owner-full-control` vordefinierten ACL geschrieben werden, automatisch dem Bucket-Eigentümer und nicht dem Objekt-Writer. Alle anderen ACL-Verhaltensweisen bleiben bestehen. Damit alle PUT-Vorgänge von Amazon S3 die vordefinierte `bucket-owner-full-control`-ACL enthalten müssen, können Sie eine Bucket-Richtlinie hinzufügen, die nur Objekt-Uploads mit dieser ACL zulässt.

Wenn Sie feststellen möchten, welche Einstellungen für die Objekteigentümerschaft auf Ihre Buckets angewendet werden, können Sie die Metriken von Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Weitere Informationen finden Sie unter Verwenden von S3 Storage Lens, um Einstellungen für die Objekteigentümerschaft zu finden.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Einstellungen für Object Ownership

Diese Tabelle zeigt die Auswirkungen, die jede Einstellung für Object Ownership auf ACLs, Objekte, Objekteigentümer und Objekt-Uploads hat.

Einstellung	Gilt für	Auswirkung auf Object Ownership	Auswirkungen auf ACLs	Hochladen akzeptiert
„Bucket-Eigentümer	Alle neuen und bestehenden Objekte	Bucket-Eigentümer besitzt jedes Objekt.	ACLs sind deaktiviert und wirken sich	Uploads mit ACLs mit vollem Zugriff

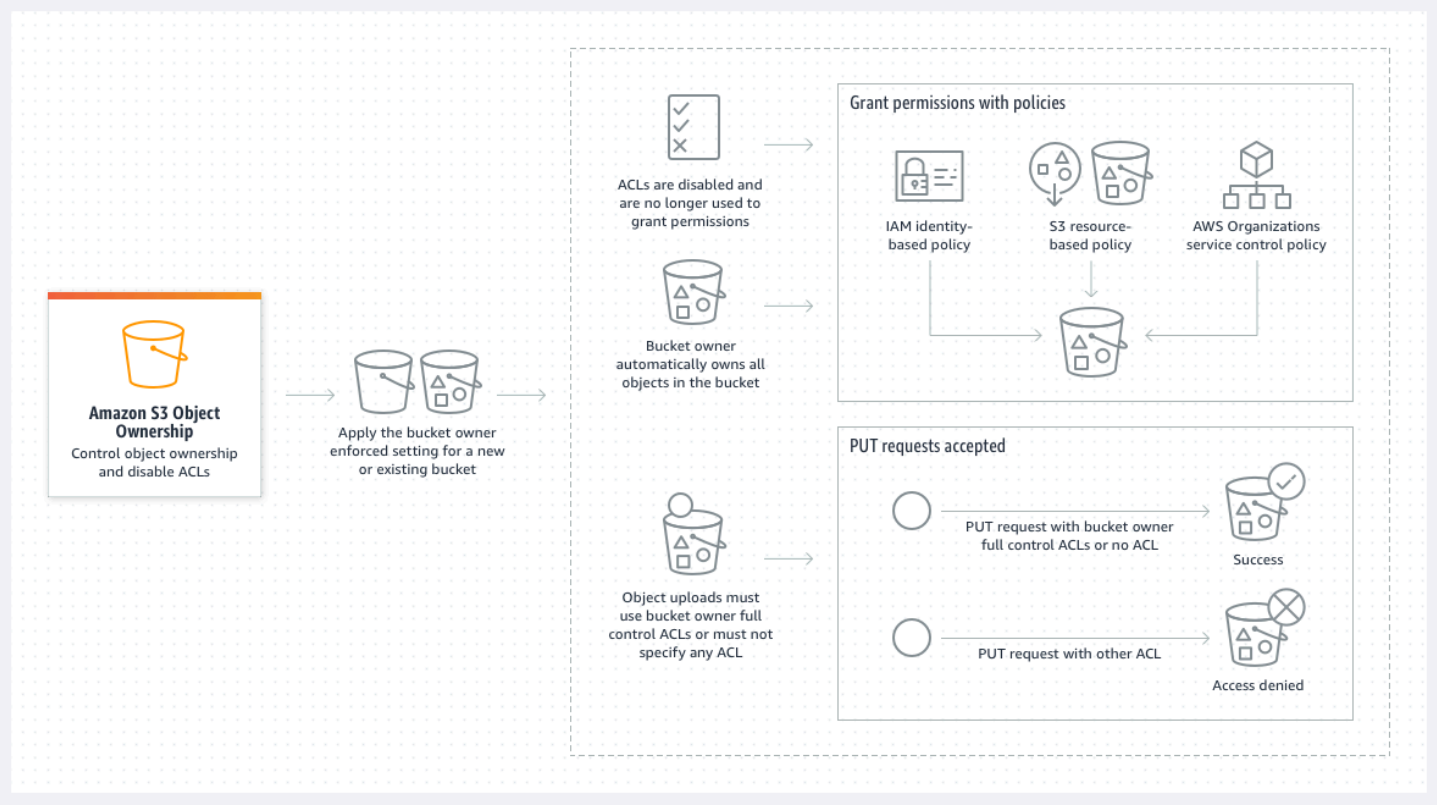
Einstellung	Gilt für	Auswirkung auf Object Ownership	Auswirkungen auf ACLs	Hochladen akzeptiert
erzwungen “ (Standard)			nicht mehr auf die Zugriffsberechtigungen für Ihren Bucket aus. Anfragen zum Festlegen oder Aktualisieren von ACLs schlagen fehl. Anfragen zum Lesen von ACLs werden jedoch unterstützt. Bucket-Eigentümer hat das volle Eigentum und die volle Kontrolle. Der Objekt-Writer hat nicht mehr das volle Eigentum und die volle Kontrolle.	des Bucket-Eigentümers oder Uploads, die keine ACL angeben

Einstellung	Gilt für	Auswirkung auf Object Ownership	Auswirkungen auf ACLs	Hochladen akzeptiert
Bucket-Eigentümer bevorzugt	Neue Objekte	Wenn ein Objekt-Upload die `bucket-owner-full-control` vordefinierte ACL beeinhaltet, gehört dem Bucket Eigentümer das Objekt. Objekte, die mit anderen ACLs hochgeladen wurden, gehören dem Schreibkonto.	ACLs können aktualisiert werden und können Berechtigungen erteilen. Wenn ein Objekt-Upload die `bucket-owner-full-control` vordefinierte ACL enthält, hat der Bucket-Eigentümer Vollzugriff und der Objekt-Writer hat keinen Vollzugriff mehr.	Alle Uploads
Objektschreiber	Neue Objekte	Der Objekt-Writer besitzt das Objekt.	ACLs können aktualisiert werden und können Berechtigungen erteilen. Der Objekt-Writer hat vollen Kontrollzugriff.	Alle Uploads

Änderungen, die durch Deaktivieren von ACLs eingeführt wurden

Wenn die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft angewendet wird, werden ACLs deaktiviert und Sie besitzen und übernehmen automatisch die volle Kontrolle über jedes Objekt im Bucket, ohne zusätzliche Aktionen auszuführen. „Bucket-Eigentümer erzwungen“ ist die Standardeinstellung für alle neu erstellten Buckets. Nachdem die Einstellung „Bucket-Eigentümer erzwungen“ angewendet wurde, sehen Sie drei Änderungen:

- Alle Bucket-ACLs und Objekt-ACLs sind deaktiviert, was Ihnen als Bucket-Eigentümer vollen Zugriff gewährt. Wenn Sie eine Lese-ACL-Anfrage für Ihren Bucket oder Objekt ausführen, werden Sie feststellen, dass nur dem Bucket-Eigentümer voller Zugriff gewährt wird.
- Sie als Bucket-Eigentümer besitzen automatisch jedes Objekt in Ihrem Bucket und haben die volle Kontrolle darüber.
- ACLs wirken sich nicht mehr auf die Zugriffsberechtigungen für Ihren Bucket aus. Daher basiert die Zugriffssteuerung für Ihre Daten auf Richtlinien wie IAM-Richtlinien, S3-Bucket-Richtlinien, VPC-Endpunktrichtlinien und SCPs von Organisationen.



Wenn Sie S3-Versioning verwenden, besitzt der Bucket-Eigentümer alle Objektversionen in Ihrem Bucket und hat die volle Kontrolle über sie. Durch das Anwenden der Einstellung „Bucket-Eigentümer erzwungen“ wird keine neue Version eines Objekts hinzugefügt.

Neue Objekte können nur dann in Ihren Bucket hochgeladen werden, wenn sie Vollzugriffs-ACLs des Bucket-Eigentümers verwenden oder keine ACL angeben. Objekt-Uploads schlagen fehl, wenn sie eine andere ACL angeben. Weitere Informationen finden Sie unter Fehlerbehebung.

Da die folgende `PutObject`-Beispieloperation mit AWS Command Line Interface (AWS CLI) die `bucket-owner-full-control` vordefinierte ACL enthält, kann das Objekt in einen Bucket mit deaktivierten ACLs hochgeladen werden.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key key-name --body path-to-file --
acl bucket-owner-full-control
```

Da der folgende `PutObject`-Vorgang keine ACL angibt, ist er auch für einen Bucket mit deaktivierten ACLs erfolgreich.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key key-name --body path-to-file
```

> **Note**
>
> Wenn andere nach dem Hochladen Zugriff auf Objekte AWS-Konten benötigen, müssen Sie diesen Konten über Bucket-Richtlinien zusätzliche Berechtigungen erteilen. Weitere Informationen finden Sie unter Beispiel-Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon-S3-Ressourcen.

ACLs erneut aktivieren

Sie können ACLs jederzeit wieder aktivieren, indem Sie von der Einstellung „Bucket-Eigentümer erzwungen“ zu einer anderen Einstellung für die Objekteigentümerschaft wechseln. Wenn Sie Objekt-ACLs für die Berechtigungsverwaltung verwendet haben, bevor Sie die Einstellung „Bucket-Eigentümer erzwungen“ angewendet haben und Sie diese Objekt-ACL-Berechtigungen nicht zu Ihrer Bucket-Richtlinie migriert haben, werden diese Berechtigungen nach dem erneuten Aktivieren von ACLs wiederhergestellt. Darüber hinaus gehören Objekte, die in den Bucket geschrieben wurden, während die „Einstellung „Bucket-Eigentümer erzwungen“ angewendet wurde, weiterhin dem Bucket-Eigentümer.

Wenn Sie beispielsweise von der Einstellung „Bucket-Eigentümer erzwungen“ zurück zur Einstellung „Objektschreiber“ wechseln, besitzen Sie als Bucket-Eigentümer nicht mehr die volle Kontrolle über Objekte, die zuvor anderen AWS-Konten gehörten. Stattdessen besitzen die Upload-Konten diese Objekte erneut. Objekte, die anderen Konten gehören, verwenden ACLs für Berechtigungen, sodass Sie keine Richtlinien verwenden können, um diesen Objekten Berechtigungen zu erteilen. Sie als Bucket-Eigentümer besitzen jedoch weiterhin alle Objekte, die in den Bucket geschrieben wurden, während die Einstellung „Bucket-Eigentümer erzwungen“ angewendet wurde. Diese Objekte gehören nicht im Besitz des Objektschreibers, auch wenn Sie ACLs erneut aktivieren.

Anweisungen zum Aktivieren und Verwalten von ACLs mithilfe der AWS Management Console, AWS Command Line Interface (CLI), REST-API oder AWS SDKs finden Sie unter Konfigurieren von ACLs.

Voraussetzungen für die Deaktivierung von ACLs

Stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind, bevor Sie ACLs für einen vorhandenen Bucket deaktivieren.

Überprüfen Sie Bucket- und Objekt-ACLs und migrieren Sie ACL-Berechtigungen

Wenn Sie ACLs deaktivieren, wirken sich die von Bucket- und Objekt-ACLs gewährten Berechtigungen nicht mehr auf den Zugriff aus. Bevor Sie ACLs deaktivieren, überprüfen Sie Ihre Bucket- und Objekt-ACLs.

Wenn Ihre Bucket-ACLs anderen Benutzern außerhalb Ihres Kontos Lese- oder Schreibberechtigungen erteilen, müssen Sie diese Berechtigungen zu Ihrer Bucket-Richtlinie migrieren, bevor Sie die Einstellung „Bucket-Eigentümer erzwungen“ anwenden können. Wenn Sie keine Bucket-ACLs migrieren, die Lese- oder Schreibzugriff außerhalb Ihres Kontos gewähren, schlägt Ihre Anfrage zum Anwenden der Einstellung „Bucket-Eigentümer erzwungen“ fehl und gibt den Fehlercode InvalidBucketAclWithObjectOwnership zurück.

Wenn Sie beispielsweise ACLs für einen Bucket deaktivieren möchten, der Serverzugriffsprotokolle empfängt, müssen Sie die Bucket-ACL-Berechtigungen für die S3-Protokollbereitstellungsgruppe zum Protokollierungsserviceprinzipal in einer Bucket-Richtlinie migrieren. Weitere Informationen finden Sie unter Gewähren von Zugriff auf die S3-Protokollbereitstellungsgruppe für die Protokollierung des Serverzugriffs.

Wenn Sie möchten, dass der Objektschreiber die volle Kontrolle über das hochgeladene Objekt behält, ist der Objektschreiber die beste Einstellung für die Objekteigentümerschaft in Ihrem Anwendungsfall. Wenn Sie den Zugriff auf der Ebene einzelner Objekte steuern möchten, ist Bucket-Eigentümer bevorzugt die beste Wahl. Diese Anwendungsfälle sind ungewöhnlich.

Informationen zum Überprüfen von ACLs und zum Migrieren von ACL-Berechtigungen auf Bucket-Richtlinien finden Sie unter Voraussetzungen für die Deaktivierung von ACLs.

Identifizieren von Anforderungen, für die eine ACL zur Autorisierung erforderlich war

Wenn Sie Amazon-S3-Anforderungen identifizieren möchten, für die ACLs zur Autorisierung erforderlich waren, können Sie den Wert `aclRequired` in den Amazon-S3-Serverzugriffsprotokollen oder AWS CloudTrail verwenden. Wenn für die Anforderung eine ACL zur Autorisierung erforderlich war oder wenn Sie PUT-Anfragen haben, die eine ACL angeben, lautet die Zeichenfolge `Yes`. Wenn keine ACLs erforderlich waren oder wenn Sie eine `bucket-owner-full-control` vordefinierte ACL festlegen oder wenn die Anforderungen von Ihrer Bucket-Richtlinie zugelassen werden, lautet die `aclRequired` Wertzeichenfolge in den Amazon S3-Serverzugriffsprotokollen „`-`“ und in fehlt sie CloudTrail. Weitere Informationen zu den erwarteten Werten für `aclRequired` finden Sie unter `aclRequired`-Werte für allgemeine Amazon-S3-Anfragen.

Wenn Sie `PutBucketAcl`- oder `PutObjectAcl`-Anfragen mit Headern haben, die ACL-basierte Berechtigungen gewähren, mit Ausnahme der vordefinierten `bucket-owner-full-control`-ACL, müssen Sie diese Header entfernen, bevor Sie ACLs deaktivieren können. Andernfalls schlagen Ihre Anfragen fehl.

Für alle anderen Anfragen, für die eine ACL zur Autorisierung erforderlich war, migrieren Sie diese ACL-Berechtigungen zu Bucket-Richtlinien. Entfernen Sie dann alle Bucket-ACLs, bevor Sie die Einstellung „Bucket-Eigentümer erzwungen“ aktivieren.

> **Note**
>
> Entfernen Sie keine Objekt-ACLs. Andernfalls verlieren Anwendungen, die bezüglich Berechtigungen auf Objekt-ACLs angewiesen sind, den Zugriff.

Wenn Sie feststellen, dass für keine Anfrage eine ACL für die Autorisierung erforderlich ist, können Sie mit der Deaktivierung der ACLs fortfahren. Weitere Informationen zur Identifizierung von Anfragen finden Sie unter Verwenden von Amazon-S3-Serverzugriffsprotokollen zur Identifizierung von Anforderungen und Identifizieren von Amazon S3-Anforderungen mit CloudTrail.

Überprüfen und aktualisieren Sie Bucket-Richtlinien, die ACL-bezogene Bedingungsschlüssel verwenden

Nachdem Sie die Einstellung „Bucket-Eigentümer erzwungen“ zum Deaktivieren von ACLs angewendet haben, können neue Objekte nur dann in Ihren Bucket hochgeladen werden, wenn die Anforderung Vollzugriffs-ACLs des Bucket-Eigentümers verwendet oder keine ACL angibt. Bevor Sie ACLs deaktivieren, überprüfen Sie Ihre Bucket-Richtlinie auf ACL-bezogene Bedingungsschlüssel.

Wenn Ihre Bucket-Richtlinie einen ACL-bezogenen Bedingungsschlüssel verwendet, um die `bucket-owner-full-control` vordefinierte ACL (z. B. `s3:x-amz-acl`) anzufordern, müssen Sie Ihre Bucket-Richtlinie nicht aktualisieren. Die folgende Bucket-Richtlinie verwendet das `s3:x-amz-acl`, um die vordefinierte `bucket-owner-full-control`-ACL für S3-`PutObject`-Anforderungen anzufordern. Diese Richtlinie erfordert immer noch, dass der Objekt-Writer die vordefinierte `bucket-owner-full-control`-ACL angibt. Buckets mit deaktivierten ACLs akzeptieren diese ACL jedoch weiterhin, sodass Anfragen weiterhin erfolgreich sind, ohne dass clientseitige Änderungen erforderlich sind.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Only allow writes to my bucket with bucket owner full control",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/ExampleUser"
                ]
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control"
                }
            }
        }
    ]
}
```

Wenn Ihre Bucket-Richtlinie jedoch einen Zustandsschlüssel im Zusammenhang mit ACL verwendet, der eine andere Zugriffssteuerungsliste erfordert, müssen Sie diesen Bedingungsschlüssel entfernen. Diese Beispiel-Bucket-Richtlinie erfordert die `public-read`-ACL für `PutObject`-S3-Anforderungen und muss daher vor dem Deaktivieren von ACLs aktualisiert werden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Only allow writes to my bucket with public read access",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/ExampleUser"                ]
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "public-read"
                }
            }
        }
    ]
}
```

Berechtigungen für Object Ownership

Um eine Einstellung für Object Ownership für einen Bucket anzuwenden, zu aktualisieren oder zu löschen, benötigen Sie die `s3:PutBucketOwnershipControls`-Berechtigung. Um die Object-Ownership-Einstellung für einen Bucket zurückzugeben, benötigen Sie die `s3:GetBucketOwnershipControls`-Berechtigung. Weitere Informationen finden Sie unter Festlegen von Object Ownership beim Erstellen eines Buckets und Anzeigen der Einstellung Object Ownership für einen S3-Bucket.

Deaktivieren von ACLs für alle neuen Buckets

Standardmäßig werden alle neuen Buckets mit der Einstellung „Bucket-Eigentümer erzwungen“ erstellt und ACLs sind deaktiviert. Wir empfehlen, ACLs deaktiviert zu lassen. Als allgemeine Regel

sollten Sie S3-ressourcenbasierte Richtlinien (Bucket-Richtlinien und Zugriffspunkt-Richtlinien) oder IAM-Richtlinien für die Zugriffssteuerung anstelle von ACLs verwenden. Richtlinien stellen eine vereinfachte und flexiblere Zugriffskontrolloption dar. Mit Bucket- und Zugriffspunktrichtlinien können Sie Regeln definieren, die allgemein für alle Anfragen an Ihre Amazon-S3-Ressourcen gelten.

Replikation und Object Ownership

Wenn Sie die S3-Replikation verwenden und die Quell- und Ziel-Buckets verschiedenen gehören AWS-Konten, können Sie ACLs deaktivieren (mit der Einstellung „Bucket-Eigentümer erzwungen“ für Object Ownership), um die Replikateigentümerschaft in den zu ändern AWS-Konto , der den Ziel-Bucket besitzt. Diese Einstellung ahmt das Verhalten der bestehenden Besitzerüberschreibung nach, ohne dass eine `s3:ObjectOwnerOverrideToBucketOwner`-Berechtigung erforderlich ist. Alle Objekte, die mit der Einstellung „Bucket-Eigentümer erzwungen“ in den Ziel-Bucket repliziert werden, gehören dem Eigentümer des Ziel-Buckets. Weitere Informationen zur Besitzerüberschreibungsoption für Replikationskonfigurationen finden Sie unter Ändern des Replikat-Eigentümers.

Einstellung von Object Ownership

Sie können eine Object-Ownership-Einstellung mithilfe der Amazon S3-Konsole, AWS CLI, AWS SDKs, Amazon S3-REST-API oder anwenden AWS CloudFormation. Die folgenden REST-API- und -AWS CLI Befehle unterstützen Object Ownership:

REST-API	AWS CLI	Beschreibung
PutBucketOwnershipControls	put-bucket-ownership-controls	Erstellt oder ändert die Einstellung Object Ownership für einen vorhandenen S3-Bucket.
CreateBucket	create-bucket	Erstellt einen Bucket mit dem `x-amz-object-ownership` -Anforderungsheader zur Angabe der Einstellung für Object Ownership.
GetBucketOwnershipControls	get-bucket-ownership-controls	Ruft die Einstellung Object Ownership für einen Amazon-S3-Bucket ab.

REST-API	AWS CLI	Beschreibung
DeleteBucketOwnershipControls	delete-bucket-ownership-controls	Löscht die Einstellung für Object Ownership für einen Amazon-S3-Bucket.

Weitere Informationen zum Anwenden und Arbeiten mit Object Ownership-Einstellungen finden Sie in den folgenden Themen.

Themen

- Voraussetzungen für die Deaktivierung von ACLs
- Festlegen von Object Ownership beim Erstellen eines Buckets
- Einstellung für Object Ownership für einen vorhandenen Bucket
- Anzeigen der Einstellung Object Ownership für einen S3-Bucket
- Deaktivieren von ACLs für alle neuen Buckets und Durchsetzung von Object Ownership
- Fehlerbehebung

Voraussetzungen für die Deaktivierung von ACLs

Wenn Ihre Bucket-ACL Zugriff außerhalb Ihres gewährt AWS-Konto, müssen Sie Ihre Bucket-ACL-Berechtigungen zu Ihrer Bucket-Richtlinie migrieren und Ihre Bucket-ACL auf die standardmäßige private ACL zurücksetzen, bevor Sie ACLs deaktivieren. Wenn Sie diese Bucket-ACLs nicht migrieren, schlägt Ihre Anfrage zum Anwenden der Einstellung „Bucket-Eigentümer erzwungen“ zum Deaktivieren von ACLs fehl und gibt den Fehlercode InvalidBucketAclWithObjectOwnership zurück. Wir empfehlen Ihnen auch, die ACL-Berechtigungen für das Objekt zu überprüfen und sie in Ihre Bucket-Richtlinie zu migrieren. Weitere Informationen zu anderen empfohlenen Voraussetzungen finden Sie unter Voraussetzungen für die Deaktivierung von ACLs.

Jede Ihrer vorhandenen Bucket- und Objekt-ACLs hat ein Äquivalent in einer IAM-Richtlinie. Die folgenden Beispiele für Bucket-Richtlinien zeigen, wie `READ`- und `WRITE`-Berechtigungen für Bucket- und Objekt-ACLs IAM-Berechtigungen zugeordnet werden. Weitere Informationen dazu, wie jede ACL in IAM-Berechtigungen übersetzt wird, finden Sie unter Mapping der ACL-Berechtigungen und Zugriffsrichtlinienberechtigungen.

Informationen zum Überprüfen und Migrieren von ACL-Berechtigungen zu Bucket-Richtlinien finden Sie in den folgenden Themen.

Themen

Beispiele für Bucket-Richtlinie

Diese Beispiel-Bucket-Richtlinien zeigen Ihnen, wie Sie `READ`- und `WRITE`-Bucket- und Objekt-ACL-Berechtigungen für einen Drittanbieter AWS-Konto in eine Bucket-Richtlinie migrieren. `READ_ACP`- und `WRITE_ACP`-ACLs sind für Richtlinien weniger relevant, da sie ACL-bezogene Berechtigungen gewähren (`s3:GetBucketAcl`, `s3:GetObjectAcl`, `s3:PutBucketAcl` und `s3:PutObjectAcl`).

Example — **`READ`**-ACL für einen Bucket

Wenn Ihr Bucket über eine `READ` ACL verfügt, die die *111122223333* Berechtigung zum Auflisten des Inhalts Ihres Buckets gewährt AWS-Konto , können Sie eine Bucket-Richtlinie schreiben, die `s3:ListBucket`-, `-s3:ListBucketVersions`, `-s3:ListBucketMultipartUploads`Berechtigungen für Ihren Bucket gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Permission to list the objects in a bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": [

                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "s3:ListBucket",
                "s3:ListBucketVersions",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        }
    ]
```

```
}
```

Example – **READ**-ACLs für jedes Objekt in einem Bucket

Wenn jedes Objekt in Ihrem Bucket über eine READ ACL verfügt, die Zugriff auf gewährt AWS-Konto *111122223333*, können Sie eine Bucket-Richtlinie schreiben, die diesem Konto für jedes Objekt in Ihrem Bucket - `s3:GetObject` und -`s3:GetObjectVersion`Berechtigungen gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Read permission for every object in a bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Dieses Beispiel-Ressourcenelement gewährt Zugriff auf ein bestimmtes Objekt.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/OBJECT-KEY"
```

Example – **WRITE**-ACL, die Berechtigungen zum Schreiben von Objekten in einen Bucket erteilt

Wenn Ihr Bucket über eine WRITE ACL verfügt, die die *111122223333* Berechtigung zum Schreiben von Objekten in Ihren Bucket gewährt AWS-Konto , können Sie eine Bucket-Richtlinie schreiben, die die `s3:PutObject` Berechtigung für Ihren Bucket gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "Permission to write objects to a bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Überprüfen und Migrieren von ACL-Berechtigungen mit der S3-Konsole

So überprüfen Sie die ACL-Berechtigungen eines Buckets

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus.
3. Wählen Sie die Registerkarte Berechtigungen.
4. Überprüfen Sie unter Zugriffskontrollliste (ACL) Ihre Bucket-ACL-Berechtigungen.

So überprüfen Sie die ACL-Berechtigungen eines Objekts

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der Ihr Objekt enthält.
3. Wählen Sie in der Liste Objekts (Objekte) den Objektnamen aus.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Überprüfen Sie unter Zugriffskontrollliste (ACL) Ihre Objekt-ACL-Berechtigungen.

So migrieren Sie ACL-Berechtigungen und aktualisieren Ihre Bucket-ACL

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie in der Liste Buckets den Namen des Buckets aus.
3. Wählen Sie auf der Registerkarte Berechtigungen unter Bucket-Richtlinie die Option Bearbeiten aus.
4. Fügen Sie im Feld Richtlinie Ihre Bucket-Richtlinie hinzu oder aktualisieren Sie sie.

   Beispiele für Bucket-Richtlinien finden Sie unter Beispiele für Bucket-Richtlinie und Beispielhafte Walkthroughs.
5. Wählen Sie Änderungen speichern aus.
6. Aktualisieren Sie Ihre Bucket ACL um Zuschüsse für andere Gruppen oder AWS-Konten zu entfernen.
7. Wenden Sie die Einstellung Bucket-Eigentümer erzwungen für die Objekteigentümerschaft an.

Verwenden der AWS CLI zum Überprüfen und Migrieren von ACL-Berechtigungen

1. Verwenden Sie den get-bucket-acl AWS CLI Befehl , um die Bucket-ACL für Ihren Bucket zurückzugeben:

   ```
   aws s3api get-bucket-acl --bucket DOC-EXAMPLE-BUCKET
   ```

   Zum Beispiel gewährt dieser Bucket-ACL `WRITE`- und `READ`-Zugriff auf ein Drittanbieter-Konto. In dieser ACL wird das Drittanbieterkonto durch die kanonische Benutzer-ID identifiziert. Um die Einstellung „Bucket-Eigentümer erzwungen“ anzuwenden und ACLs zu deaktivieren, müssen Sie diese Berechtigungen für das Drittanbieterkonto zu einer Bucket-Richtlinie migrieren.

   ```
   {
       "Owner": {
           "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
           "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
       },
       "Grants": [
           {
               "Grantee": {
                   "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                   "ID":
    "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                   "Type": "CanonicalUser"
               },
               "Permission": "FULL_CONTROL"
   ```

```
        },
        {
            "Grantee": {
                "DisplayName": "THIRD-PARTY-EXAMPLE-ACCOUNT",
                "ID":
 "72806de9d1ae8b171cca9e2494a8d1335dfced4ThirdPartyAccountCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "READ"
        },
        {
            "Grantee": {
                "DisplayName": "THIRD-PARTY-EXAMPLE-ACCOUNT",
                "ID":
 "72806de9d1ae8b171cca9e2494a8d1335dfced4ThirdPartyAccountCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "WRITE"
        }
    ]
}
```

Weitere Beispiel-ACLs finden Sie unter Beispielhafte Walkthroughs.

2. Migrieren Sie Ihre Bucket-ACL-Berechtigungen auf eine Bucket-Richtlinie:

   In diesem Beispiel gewährt Bucket-Richtlinien `s3:PutObject` und `s3:ListBucket`Berechtigungen für ein Drittanbieterkonto. In der Bucket-Richtlinie wird das Drittanbieterkonto durch die - AWS-Konto ID () identifiziert*111122223333*.

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PolicyForCrossAccountAllowUpload",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
```

```
            },
            "Action": [
                "s3:PutObject",
                "s3:ListBucket"
            ],
            "Resource": [
             "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
             "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Weitere Beispiele für Bucket-Richtlinien finden Sie unter Beispiele für Bucket-Richtlinie und Beispielhafte Walkthroughs.

3. Verwenden Sie den get-object-acl AWS CLI Befehl , um die ACL für ein bestimmtes Objekt zurückzugeben.

```
aws s3api get-object-acl --bucket DOC-EXAMPLE-BUCKET --key EXAMPLE-OBJECT-KEY
```

4. Migrieren Sie bei Bedarf Objekt-ACL-Berechtigungen auf Ihre Bucket-Richtlinie.

   Dieses Beispiel-Ressourcenelement gewährt Zugriff auf ein bestimmtes Objekt in einer Bucket-Richtlinie.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/EXAMPLE-OBJECT-KEY"
```

5. Setzen Sie die ACL für Ihren Bucket auf die Standard-ACL zurück.

```
aws s3api put-bucket-acl --bucket DOC-EXAMPLE-BUCKET --acl private
```

6. Wenden Sie die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft an.

Beispielhafte Walkthroughs

Die folgenden Beispiele zeigen, wie Sie ACL-Berechtigungen für bestimmte Anwendungsfälle zu Bucket-Richtlinien migrieren.

Themen

- Gewähren von Zugriff auf die S3-Protokollbereitstellungsgruppe für die Protokollierung des Serverzugriffs

- Öffentlichen Lesezugriff auf die Objekte in einem Bucket gewähren
- Amazon ElastiCache for Redis Zugriff auf Ihren S3-Bucket gewähren

Gewähren von Zugriff auf die S3-Protokollbereitstellungsgruppe für die Protokollierung des Serverzugriffs

Wenn Sie die Einstellung „Von Bucket-Besitzer erzwungen“ zur Deaktivierung von ACLs für einen Serverzugriffsprotokollierungs-Ziel-Bucket (Ziel-Bucket) verwenden möchten, müssen Sie die Bucket-ACL-Berechtigungen für die S3-Protokollbereitstellungsgruppe zum Prinzipal des Protokollierungsservices (`logging.s3.amazonaws.com`) in einer Bucket-Richtlinie migrieren. Weitere Informationen zu Berechtigungen für die Protokollzustellung finden Sie unter Berechtigungen für Protokollbereitstellung.

Diese Bucket-ACL gewährt `WRITE`- und `READ_ACP`-Zugriff auf die S3-Logbereitstellungsgruppe:

```
{
    "Owner": {
        "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "Type": "CanonicalUser",
                "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/s3/LogDelivery"
            },
            "Permission": "WRITE"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/s3/LogDelivery"
```

```
            },
            "Permission": "READ_ACP"
        }
    ]
}
```

So migrieren Sie Bucket-ACL-Berechtigungen für die S3-Protokollbereitstellungsgruppe zum Protokollierungsdienstprinzipal in einer Bucket-Richtlinie

1. Fügen Sie Ihrem Ziel-Bucket die folgende Bucket-Richtlinie hinzu. wobei Sie die Beispielwerte ersetzen.

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:      {
    {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ServerAccessLogsPolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/EXAMPLE-LOGGING-PREFIX*",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::SOURCE-BUCKET-NAME"
                },
                "StringEquals": {
                    "aws:SourceAccount": "SOURCE-AWS-ACCOUNT-ID"
                }
            }
        }
    ]
}
```

2. Setzen Sie die ACL für Ihren Ziel-Bucket auf die Standard-ACL zurück.

```
aws s3api put-bucket-acl --bucket DOC-EXAMPLE-BUCKET --acl private
```

3. Wenden Sie die Einstellung „Von Bucket-Besitzer erzwungen“ für den Objektbesitz auf Ihren Ziel-Bucket an.

Öffentlichen Lesezugriff auf die Objekte in einem Bucket gewähren

Wenn Ihre Objekt-ACLs öffentlichen Lesezugriff auf alle Objekte in Ihrem Bucket gewähren, können Sie diese ACL-Berechtigungen auf eine Bucket-Richtlinie migrieren.

Diese Objekt-ACL gewährt öffentlichen Lesezugriff auf ein Objekt in einem Bucket:

```
{
    "Owner": {
        "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

So migrieren Sie Berechtigungen für öffentliche Lese-ACL auf eine Bucket-Richtlinie

1. Um öffentlichen Lesezugriff auf alle Objekte in Ihrem Bucket zu gewähren, fügen Sie die folgende Bucket-Richtlinie hinzu und ersetzen Sie die Beispielwerte.

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

Um öffentlichen Zugriff auf ein bestimmtes Objekt in einer Bucket-Richtlinie zu gewähren, verwenden Sie das folgende Format für das `Resource`-Element.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/OBJECT-KEY"
```

Um öffentlichen Zugriff auf alle Objekte mit einem bestimmten Präfix zu gewähren, verwenden Sie das folgende Format für das `Resource`-Element.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/PREFIX/*"
```

2. Wenden Sie die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft an.

Amazon ElastiCache for Redis Zugriff auf Ihren S3-Bucket gewähren

Sie können Ihr ElastiCache für Redis-Backup in einen S3-Bucket exportieren, sodass Sie von außerhalb auf das Backup zugreifen können ElastiCache. S3 Um Ihr Backup in einen S3-Bucket zu exportieren, müssen Sie Berechtigungen zum Kopieren eines Snapshots in den Bucket erteilen ElastiCache. Wenn Sie Berechtigungen für ElastiCache in einer Bucket-ACL erteilt haben, müssen Sie diese Berechtigungen zu einer Bucket-Richtlinie migrieren, bevor Sie die Einstellung „Bucket-

Eigentümer erzwungen“ anwenden, um ACLs zu deaktivieren. Weitere Informationen finden Sie unter ElastiCache Zugriff auf Ihren Amazon S3-Bucket gewähren im Amazon- ElastiCache Benutzerhandbuch.

Das folgende Beispiel zeigt die Bucket-ACL-Berechtigungen, die Berechtigungen für erteilen ElastiCache.

```
{
    "Owner": {
        "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "DisplayName": "aws-scs-s3-readonly",
                "ID":
"540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353",
                "Type": "CanonicalUser"
            },
            "Permission": "READ"
        },
        {
            "Grantee": {
                "DisplayName": "aws-scs-s3-readonly",
                "ID":
"540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353",
                "Type": "CanonicalUser"
            },
            "Permission": "WRITE"
        },
        {
            "Grantee": {
                "DisplayName": "aws-scs-s3-readonly",
```

```
                "ID":
 "540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353",
                "Type": "CanonicalUser"
            },
            "Permission": "READ_ACP"
        }
    ]
}
```

So migrieren Sie Bucket-ACL-Berechtigungen für ElastiCache für Redis zu einer Bucket-Richtlinie

1. Fügen Sie Ihrem Bucket die folgende Bucket-Richtlinie hinzu und ersetzen Sie die Beispielwerte.

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:
"Id": "Policy15397346",
    "Statement": [
        {
            "Sid": "Stmt15399483",
            "Effect": "Allow",
            "Principal": {
                "Service": "Region.elasticache-snapshot.amazonaws.com"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

2. Setzen Sie die ACL für Ihren Bucket auf die Standard-ACL zurück:

```
aws s3api put-bucket-acl --bucket DOC-EXAMPLE-BUCKET --acl private
```

3. Wenden Sie die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft an.

Festlegen von Object Ownership beim Erstellen eines Buckets

Wenn Sie einen Bucket erstellen, können Sie S3 Object Ownership konfigurieren. Informationen zum Festlegen von Object Ownership für einen vorhandenen Bucket finden Sie unter Einstellung für Object Ownership für einen vorhandenen Bucket.

S3 Object Ownership ist eine Einstellung auf Amazon-S3-Bucket-Ebene, mit der Sie Zugriffskontrolllisten (ACLs) deaktivieren und das Eigentum an jedem Objekt in Ihrem Bucket übernehmen können, wodurch die Zugriffsverwaltung für in Amazon S3 gespeicherte Daten vereinfacht wird. Standardmäßig ist S3 Object Ownership auf die Einstellung „Von Bucket-Besitzer erzwungen“ festgelegt und ACLs sind für neue Buckets deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer jedes Objekt im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen.

Object Ownership verfügt über drei Einstellungen, mit denen Sie die Eigentümerschaft von Objekten, die in Ihren Bucket hochgeladen werden, steuern und ACLs deaktivieren oder aktivieren können:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Berechtigungen für Daten im S3-Bucket. Der Bucket verwendet Richtlinien, um die Zugriffssteuerung zu definieren.

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.
- Object Writer – Das AWS-Konto , das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

Berechtigungen: Um die Einstellung Bucket-Eigentümer erzwungen oder Bucket-Eigentümer bevorzugt anzuwenden, müssen Sie über die folgenden Berechtigungen verfügen: `s3:CreateBucket` und `s3:PutBucketOwnershipControls`. Wenn Sie einen Bucket mit aktivierter Einstellung Object writer (Objektschreiber), sind keine zusätzlichen Berechtigungen erforderlich. Weitere Informationen zu Amazon S3-Berechtigungen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

> **Important**
>
> Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr, und wir empfehlen Ihnen, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Mit Object Ownership können Sie ACLs deaktivieren und sich auf Richtlinien für die Zugriffssteuerung verlassen. Wenn Sie ACLs deaktivieren, können Sie einen Bucket mit Objekten verwalten, die von verschiedenen AWS Konten hochgeladen wurden. Sie als Bucket-Eigentümer besitzen alle Objekte im Bucket und können den Zugriff darauf mithilfe von Richtlinien verwalten.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Anschließend wird die Seite Bucket erstellen geöffnet.
4. Geben Sie unter Bucket Name (Bucket-Name) einen Namen für den Bucket ein.

   Der Bucket-Name ...:

   - Muss innerhalb einer Partition einzigartig sein. Eine Partition ist eine Gruppierung von Regionen. AWS verfügt derzeit über drei Partitionen: `aws` (Standardregionen), `aws-cn` (China-Regionen) und `aws-us-gov` (AWS GovCloud (US) Regions).
   - zwischen 3 und 63 Zeichen lang sein,
   - Darf nur aus Kleinbuchstaben, Zahlen, Punkten (.) und Bindestrichen (-) bestehen. Aus Gründen der besten Kompatibilität empfehlen wir, Punkte (.) in Bucket-Namen zu vermeiden, mit Ausnahme von Buckets, die nur für statisches Website-Hosting verwendet werden.

- Muss mit einer Zahl oder einem Buchstaben beginnen und enden.

Der Name eines einmal erstellter Buckets kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

> **Important**
>
> Vermeiden Sie, vertrauliche Informationen, wie Kontonummern, in den Bucket-Namen einzubeziehen. Der Bucket-Name wird in der URL angezeigt, die auf die Objekte im Bucket verweist.

5. Wählen Sie für Region die aus, AWS-Region in der sich der Bucket befinden soll.

   Wählen Sie eine Region in der Nähe aus, um Latenz und Kosten gering zu halten und behördliche Vorschriften zu erfüllen. In einer Region gespeicherte Objekte verbleiben so lange in der Region, bis sie explizit in eine andere Region verschoben werden. Eine Liste von Amazon S3 AWS-Regionen finden Sie unter -AWS-Service Endpunkte im Allgemeine Amazon Web Services-Referenz.

6. Wählen Sie unter Object Ownership eine der folgenden Einstellungen aus, um ACLs zu deaktivieren oder zu aktivieren und den Besitz von Objekten zu steuern, die in Ihren Bucket hochgeladen wurden:

   Deaktivierte ACLs

   - Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Zugriffsberechtigungen für Daten im S3-Bucket. Der Bucket verwendet ausschließlich Richtlinien, um die Zugriffssteuerung zu definieren.

     Standardmäßig sind ACLs deaktiviert. Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.

  Wenn Sie die Einstellung Bucket-Eigentümer bevorzugt anwenden, damit alle Amazon-S3-Uploads die von `bucket-owner-full-control` vordefinierte ACL enthalten, können Sie eine Bucket-Richtlinie hinzufügen, die nur Objekt-Uploads zulässt, die diese ACL verwenden.
- Object Writer – Das AWS-Konto , das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

> **Note**
>
> Die Standardeinstellung ist Bucket-Eigentümer erzwungen. Um die Standardeinstellung anzuwenden und ACLs deaktiviert zu lassen, ist nur die `s3:CreateBucket`-Berechtigung erforderlich. Sie müssen über die `s3:PutBucketOwnershipControls`-Berechtigung verfügen, um ACLs zu aktivieren.

7. Wählen Sie unter Einstellungen "Öffentlichen Zugriff beschränken" für diesen Bucket die Einstellungen zum Beschränken des öffentlichen Zugriffs aus, die Sie auf den Bucket anwenden möchten.

   Alle vier Einstellungen zum Blockieren des öffentlichen Zugriffs sind standardmäßig aktiviert. Es wird empfohlen, alle Einstellungen aktiviert zu lassen, es sei denn, Sie wissen, dass Sie eine oder mehrere dieser Einstellungen für Ihren Anwendungsfall deaktivieren müssen. Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

   > **Note**
   >
   > Zum Aktivieren aller Einstellungen zum Blockieren des öffentlichen Zugriffs ist nur die `s3:CreateBucket`-Berechtigung erforderlich. Wenn Sie eine der Einstellungen zum Blockieren des öffentlichen Zugriffs deaktivieren möchten, benötigen Sie die `s3:PutBucketPublicAccessBlock`-Berechtigung.

8. (Optional) Unter Bucket Versioning (Bucket-Versionsverwaltung) können Sie auswählen, ob Sie Varianten von Objekten in Ihrem Bucket beibehalten möchten. Weitere Informationen über das Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

   Wenn Sie die Versionsverwaltung in Ihrem Bucket deaktivieren oder aktivieren möchten, wählen Sie entweder Disable (Deaktivieren) oder Enable (Aktivieren) aus.

9. (Optional) Unter Tags können Sie auswählen, ob Sie Ihrem Bucket Tags hinzufügen möchten. Tags sind Schlüssel-Wert-Paare, die zur Kategorisierung von Speicher verwendet werden.

   Wenn Sie ein Bucket-Tag hinzuzufügen, geben Sie einen Key (Schlüssel) und optional einen Value (Wert) ein. Wählen Sie dann Add Tag (Tag hinzufügen) aus.

10. Wählen Sie unter Default encryption (Standard-Verschlüsselung) Edit (Bearbeiten) aus.

11. Wählen Sie eine der folgenden Optionen unter Verschlüsselungstyp aus, um die Standardverschlüsselung zu konfigurieren:

    - Von Amazon S3 verwalteter Schlüssel (SSE-S3)
    - AWS Key Management Service -Schlüssel (SSE-KMS)

    > **Important**
    >
    > Wenn Sie die Option SSE-KMS für die Standardverschlüsselung verwenden, unterliegen Sie den Kontigenten der Anforderungen pro Sekunde (RPS) von AWS KMS. Weitere Informationen zu AWS KMS Kontingenten und zum Anfordern einer Kontingenterhöhung finden Sie unter Kontingente im AWS Key Management Service - Entwicklerhandbuch.

    Buckets und neue Objekte werden mit serverseitiger Verschlüsselung verschlüsselt. Dabei ist ein Von Amazon S3 verwalteter Schlüssel die Grundebene der Verschlüsselungskonfiguration. Weitere Informationen zur Standardverschlüsselung finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

    Weitere Informationen zur Datenverschlüsselung mit der serverseitigen Amazon-S3-Verschlüsselung finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

12. Wenn Sie AWS Key Management Service -Schlüssel (SSE-KMS) ausgewählt haben, gehen Sie wie folgt vor:

a. Geben Sie unter AWS KMS -Schlüssel Ihren KMS-Schlüssel auf eine der folgenden Arten an:

- Um aus einer Liste der verfügbaren KMS-Schlüssel auszuwählen, wählen Sie Aus Ihrem AWS KMS keys auswählen und wählen Sie Ihren KMS-Schlüssel aus der Liste der verfügbaren Schlüssel aus.

  Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS -Schlüssel im Entwicklerhandbuch zu AWS Key Management Service .
- Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS key -ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
- Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

  Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

> **Important**
>
> Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket verfügbar sind. Die Amazon-S3-Konsole führt nur die ersten 100 KMS-Schlüssel auf, die in derselben Region wie der Bucket verfügbar sind. Wenn Sie einen KMS-Schlüssel verwenden möchten, der nicht aufgeführt ist, müssen Sie den KMS-Schlüssel-ARN eingeben. Wenn Sie einen KMS-Schlüssel verwenden möchten, der sich im Besitz eines anderen Kontos befindet, müssen Sie über die Berechtigung zum Verwenden des Schlüssels verfügen und Sie müssen den KMS-Schlüssel-ARN eingeben. Weitere Informationen zu kontoübergreifenden Berechtigungen für KMS-Schlüssel finden Sie unter Erstellen von KMS-Schlüsseln, die von anderen Konten verwendet werden können im Entwicklerhandbuch zu AWS Key Management Service . Weitere Informationen zu SSE-KMS finden Sie unter Angeben der serverseitigen Verschlüsselung mit AWS KMS -(SSE-KMS).
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer

Verschlüsselung und keine asymmetrischen KMS-Schlüssel. Weitere Informationen finden Sie unter Erkennen von symmetrischen und asymmetrischen KMS-Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service .

Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch. Weitere Informationen zur Verwendung von AWS KMS mit Amazon S3 finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

b. Wenn Sie Ihren Bucket für die Verwendung der Standardverschlüsselung mit SSE-KMS konfigurieren, können Sie auch S3-Bucket-Schlüssel aktivieren. S3-Bucket-Schlüssel senken die Verschlüsselungskosten, indem der Anforderungsverkehr von Amazon S3 zu verringert wird AWS KMS. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.

   Um S3-Bucket-Schlüssel zu verwenden, wählen Sie unter Bucket Key (Bucket-Schlüssel) die Option Enable (Aktivieren).

13. (Optional) Wenn Sie die S3-Objektsperre aktivieren möchten, gehen Sie wie folgt vor:

    a. Wählen Sie Erweiterte Einstellungen aus.

    > **Important**
    >
    > Durch Aktivieren der Objektsperre wird auch die Versioning für den Bucket aktiviert. Nach dem Aktivieren müssen Sie die Standardeinstellungen für die Objektsperre im Hinblick auf die (rechtliche) Aufbewahrung konfigurieren, um neue Objekte vor dem Löschen oder Überschreiben zu schützen.

    b. Wenn Sie die Objektsperre aktivieren möchten, wählen Sie Enable (Aktivieren) aus, lesen Sie die angezeigte Warnung und bestätigen Sie sie.

    Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

> **Note**
>
> Wenn Sie einen Bucket mit aktivierter Objektsperre erstellen möchten, benötigen Sie die folgenden Berechtigungen: `s3:CreateBucket`, `s3:PutBucketVersioning` und `s3:PutBucketObjectLockConfiguration`.

14. Wählen Sie Bucket erstellen aus.

Verwenden der AWS CLI

Um Object Ownership beim Erstellen eines neuen Buckets festzulegen, verwenden Sie den `create-bucket` AWS CLI Befehl mit dem `--object-ownership` Parameter .

In diesem Beispiel wird die Einstellung „Bucket-Eigentümer erzwungen“ für einen neuen Bucket mithilfe der AWS CLI angewendet:

```
aws s3api create-bucket --bucket  DOC-EXAMPLE-BUCKET --region us-east-1 --object-
ownership BucketOwnerEnforced
```

> **Important**
>
> Wenn Sie Object Ownership nicht festlegen, wenn Sie einen Bucket mithilfe der erstellen AWS CLI, ist die Standardeinstellung `ObjectWriter` (ACLs aktiviert).

Verwenden des AWS SDK for Java

In diesem Beispiel wird die Einstellung „Bucket-Eigentümer erzwungen“ für einen neuen Bucket mithilfe von AWS SDK for Java festgelegt:

```
    // Build the ObjectOwnership for CreateBucket
    CreateBucketRequest createBucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .objectOwnership(ObjectOwnership.BucketOwnerEnforced)
            .build()

     // Send the request to Amazon S3
     s3client.createBucket(createBucketRequest);
```

Verwenden von AWS CloudFormation

Informationen zur Verwendung der `AWS::S3::Bucket` AWS CloudFormation Ressource zum Festlegen von Object Ownership beim Erstellen eines neuen Buckets finden Sie OwnershipControls unter in AWS::S3::Bucket im AWS CloudFormation -Benutzerhandbuch.

Verwenden der REST-API

Wenn Sie die Einstellung „Bucket-Eigentümer erzwungen“ für S3 Object Ownership anwenden möchten, verwenden Sie die API-Operation `CreateBucket`, wobei der `x-amz-object-ownership`-Anforderungsheader auf `BucketOwnerEnforced` festgelegt ist. Weitere Informationen und Beispiele finden Sie unter CreateBucket in der API-Referenz zu Amazon Simple Storage Service.

Nächste Schritte: Nach der Anwendung der Einstellungen „Von Bucket-Besitzer erzwungen“ oder „Von Bucket-Besitzer bevorzugt“ auf den Objektbesitz können Sie die folgenden Schritte ausführen:

- Bucket-Eigentümer erzwungen – Erfordert, dass alle neuen Buckets mit deaktivierten ACLs erstellt werden, indem eine IAM- oder Organisationsrichtlinie verwendet wird.
- Bucket-Eigentümer bevorzugt- Fügen Sie eine S3-Bucket-Richtlinie hinzu, um die `bucket-owner-full-control` vordefinierte ACL für alle Objekt-Uploads in Ihren Bucket anzufordern.

Einstellung für Object Ownership für einen vorhandenen Bucket

Sie können S3 Object Ownership für einen vorhandenen S3-Bucket konfigurieren. Zum Anwenden von Objekt-Ownership beim Erstellen eines Buckets finden Sie unter Festlegen von Object Ownership beim Erstellen eines Buckets.

S3 Object Ownership ist eine Einstellung auf Amazon-S3-Bucket-Ebene, mit der Sie Zugriffskontrolllisten (ACLs) deaktivieren und das Eigentum an jedem Objekt in Ihrem Bucket übernehmen können, wodurch die Zugriffsverwaltung für in Amazon S3 gespeicherte Daten vereinfacht wird. Standardmäßig ist S3 Object Ownership auf die Einstellung „Von Bucket-Besitzer erzwungen“ festgelegt und ACLs sind für neue Buckets deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer jedes Objekt im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen.

Object Ownership verfügt über drei Einstellungen, mit denen Sie die Eigentümerschaft von Objekten, die in Ihren Bucket hochgeladen werden, steuern und ACLs deaktivieren oder aktivieren können:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Berechtigungen für Daten im S3-Bucket. Der Bucket verwendet Richtlinien, um die Zugriffssteuerung zu definieren.

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.
- Object Writer – Das AWS-Konto , das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

Voraussetzungen: Bevor Sie die Einstellung „Bucket-Eigentümer erzwungen“ anwenden, um ACLs zu deaktivieren, müssen Sie Bucket-ACL-Berechtigungen zu Bucket-Richtlinien migrieren und Ihre Bucket-ACLs auf die standardmäßige private ACL zurücksetzen. Wir empfehlen außerdem, Objekt-ACL-Berechtigungen zu Bucket-Richtlinien zu migrieren und Bucket-Richtlinien zu bearbeiten, die andere ACLs als ACLs mit Vollzugriff des Bucket-Eigentümers erfordern. Weitere Informationen finden Sie unter Voraussetzungen für die Deaktivierung von ACLs.

Berechtigungen: Um diesen Vorgang zu verwenden, müssen Sie die `s3:PutBucketOwnershipControls`-Berechtigung haben. Weitere Informationen zu Amazon S3-Berechtigungen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, auf den Sie eine Einstellung für S3 Object Ownership anwenden möchten.
3. Wählen Sie die Registerkarte Berechtigungen.
4. Wählen Sie unter Object Ownership die Option Edit (Bearbeiten).
5. Wählen Sie unter Object Ownership eine der folgenden Einstellungen aus, um ACLs zu deaktivieren oder zu aktivieren und den Besitz von Objekten zu steuern, die in Ihren Bucket hochgeladen wurden:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen – ACLs sind deaktiviert, und der Bucket-Eigentümer besitzt automatisch und hat die volle Kontrolle über jedes Objekt im Bucket. ACLs haben keine Auswirkungen mehr auf Berechtigungen für Daten im S3-Bucket. Der Bucket verwendet Richtlinien, um die Zugriffssteuerung zu definieren.

  Informationen dazu, wie Sie erzwingen, dass alle neuen Buckets mit deaktivierten ACLs mithilfe von IAM- oder - AWS Organizations Richtlinien erstellt werden, finden Sie unter Deaktivieren von ACLs für alle neuen Buckets (Bucket-Eigentümer erzwungen).

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.

  Wenn Sie die Einstellung Bevorzugter Bucket-Eigentümer anwenden, damit alle Amazon-S3-Uploads die von `bucket-owner-full-control` vordefinierte ACL enthalten, können Sie eine Bucket-Richtlinie hinzufügen, die nur Objekt-Uploads zulässt, die diese ACL verwenden.
- Object Writer – Das AWS-Konto , das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

6. Wählen Sie Speichern.

Verwenden der AWS CLI

Um eine Einstellungen für Object Ownership für einen vorhandenen Bucket anzuwenden, verwenden Sie den `put-bucket-ownership-controls`-Befehl mit dem `--ownership-controls`-Parameter. Gültige Werte für die Eigentümerschaft sind `BucketOwnerEnforced`, `BucketOwnerPreferred` oder `ObjectWriter`.

In diesem Beispiel wird die Einstellung „Bucket-Eigentümer erzwungen“ für einen vorhandenen Bucket mithilfe der AWS CLI angewendet:

```
aws s3api put-bucket-ownership-controls --bucket DOC-EXAMPLE-BUCKET --ownership-
controls="Rules=[{ObjectOwnership=BucketOwnerEnforced}]"
```

Informationen zu `put-bucket-ownership-controls` finden Sie unter put-bucket-ownership-controls im AWS Command Line Interface -Benutzerhandbuch.

Verwenden des AWS SDK for Java

In diesem Beispiel gilt die `BucketOwnerEnforced`-Einstellung für Object Ownership für einen vorhandenen Bucket mit AWS SDK for Java:

```
        // Build the ObjectOwnership for BucketOwnerEnforced
        OwnershipControlsRule rule = OwnershipControlsRule.builder()
               .objectOwnership(ObjectOwnership.BucketOwnerEnforced)
               .build();

        OwnershipControls ownershipControls = OwnershipControls.builder()
                  .rules(rule)
                  .build()

         // Build the PutBucketOwnershipControlsRequest
         PutBucketOwnershipControlsRequest putBucketOwnershipControlsRequest =
               PutBucketOwnershipControlsRequest.builder()
                       .bucket(BUCKET_NAME)
                       .ownershipControls(ownershipControls)
                       .build();

         // Send the request to Amazon S3
         s3client.putBucketOwnershipControls(putBucketOwnershipControlsRequest);
```

Verwenden von AWS CloudFormation

Informationen AWS CloudFormation zur Verwendung von zum Anwenden einer Object-Ownership-Einstellung für einen vorhandenen Bucket finden Sie AWS::S3::Bucket OwnershipControls unter im AWS CloudFormation -Benutzerhandbuch.

Verwenden der REST-API

Um die REST-API zum Anwenden einer Object-Ownership-Einstellung auf einen vorhandenen S3-Bucket zu verwenden, verwenden Sie `PutBucketOwnershipControls`. Weitere Informationen finden Sie unter PutBucketOwnershipControls in der API-Referenz zu Amazon Simple Storage Service.

Nächste Schritte: Nach der Anwendung der Einstellungen „Von Bucket-Besitzer erzwungen“ oder „Von Bucket-Besitzer bevorzugt“ auf den Objektbesitz können Sie die folgenden Schritte ausführen:

- Bucket-Eigentümer erzwungen – Erfordert, dass alle neuen Buckets mit deaktivierten ACLs erstellt werden, indem eine IAM- oder Organisationsrichtlinie verwendet wird.
- Bucket-Eigentümer bevorzugt- Fügen Sie eine S3-Bucket-Richtlinie hinzu, um die `bucket-owner-full-control` vordefinierte ACL für alle Objekt-Uploads in Ihren Bucket anzufordern.

Anzeigen der Einstellung Object Ownership für einen S3-Bucket

S3 Object Ownership ist eine Einstellung auf Amazon-S3-Bucket-Ebene, mit der Sie Zugriffskontrolllisten (ACLs) deaktivieren und das Eigentum an jedem Objekt in Ihrem Bucket übernehmen können, wodurch die Zugriffsverwaltung für in Amazon S3 gespeicherte Daten vereinfacht wird. Standardmäßig ist S3 Object Ownership auf die Einstellung „Von Bucket-Besitzer erzwungen“ festgelegt und ACLs sind für neue Buckets deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer jedes Objekt im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen.

Object Ownership verfügt über drei Einstellungen, mit denen Sie die Eigentümerschaft von Objekten, die in Ihren Bucket hochgeladen werden, steuern und ACLs deaktivieren oder aktivieren können:

Deaktivierte ACLs

- Bucket-Eigentümer erzwungen (Standard) – ACLs sind deaktiviert und der Bucket-Eigentümer besitzt automatisch jedes Objekt im Bucket und hat die volle Kontrolle darüber. ACLs haben keine Auswirkungen mehr auf Berechtigungen für Daten im S3-Bucket. Der Bucket verwendet Richtlinien, um die Zugriffssteuerung zu definieren.

Aktivierte ACLs

- Bucket-Eigentümer bevorzugt – Der Bucket-Eigentümer besitzt und hat die volle Kontrolle über neue Objekte, die andere Konten mit der `bucket-owner-full-control`-vordefinierten ACL.
- Object Writer – Das AWS-Konto , das ein Objekt hochlädt, besitzt das Objekt, hat die volle Kontrolle darüber und kann anderen Benutzern über ACLs Zugriff darauf gewähren.

Sie können die S3 Object Ownership-Einstellungen für einen Amazon-S3-Bucket anzeigen. Informationen zum Festlegen von Object Ownership für einen neuen Bucket finden Sie unter Festlegen von Object Ownership beim Erstellen eines Buckets. Informationen zum Festlegen von

Object Ownership für einen vorhandenen Bucket finden Sie unter Einstellung für Object Ownership für einen vorhandenen Bucket.

Berechtigungen Um diesen Vorgang verwenden zu können, müssen Sie die `s3:GetBucketOwnershipControls`-Berechtigung haben. Weitere Informationen zu Amazon S3-Berechtigungen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, auf den Sie eine Einstellung für Object Ownership anwenden möchten.
3. Wählen Sie die Registerkarte Berechtigungen.
4. Unter Object Ownership können Sie die Objekteigentümerschafts-Einstellungen für Ihren Bucket anzeigen.

Verwenden der AWS CLI

Verwenden Sie den get-bucket-ownership-controls AWS CLI Befehl , um die Einstellung S3 Object Ownership für einen S3-Bucket abzurufen.

```
aws s3api get-bucket-ownership-controls --bucket DOC-EXAMPLE-BUCKET
```

Verwenden der REST-API

Verwenden Sie die `GetBucketOwnershipControls`-API-Operation, um die Object-Ownership-Einstellung für einen S3-Bucket abzurufen. Weitere Informationen finden Sie unter GetBucketOwnershipControls.

Deaktivieren von ACLs für alle neuen Buckets und Durchsetzung von Object Ownership

Wir empfehlen, ACLs für Ihre Amazon-S3-Buckets zu deaktivieren. Wenden Sie dazu die Einstellung „Bucket-Eigentümer erzwungen“ für S3 Object Ownership an. Wenn Sie diese Einstellung anwenden, sind ACLs deaktiviert und Sie besitzen automatisch alle Objekte in Ihrem Bucket und haben die volle Kontrolle über sie. Um zu verlangen, dass alle neuen Buckets mit deaktivierten ACLs erstellt werden,

verwenden Sie AWS Identity and Access Management (IAM)-Richtlinien oder AWS Organizations Service-Kontrollrichtlinien (SCPs), wie im nächsten Abschnitt beschrieben.

Um den Objektbesitz für neue Objekte zu erzwingen, ohne ACLs zu deaktivieren, können Sie die bevorzugte Einstellung des Bucket-Eigentümers anwenden. Wenn Sie diese Einstellung anwenden, empfehlen wir Ihnen dringend, Ihre Bucket-Richtlinie so zu aktualisieren, dass die vordefinierte `bucket-owner-full-control`-ACL für alle PUT-Anforderungen an Ihren Bucket erforderlich ist. Aktualisieren Sie unbedingt auch Ihre Clients, damit sie die vordefinierte `bucket-owner-full-control`-ACL von anderen Konten an Ihren Bucket senden.

Themen

Deaktivieren von ACLs für alle neuen Buckets (Bucket-Eigentümer erzwungen)

Die folgende IAM-Beispielrichtlinie verweigert die `s3:CreateBucket`-Berechtigung für einen bestimmten IAM-Benutzer oder eine bestimmte IAM-Rolle, es sei denn, die Einstellung „Bucket-Eigentümer erzwungen“ wird für die Objekteigentümerschaft angewendet. Das Schlüssel-Wert-Paar im `Condition`-Block gibt `s3:x-amz-object-ownership` als Schlüssel und die `BucketOwnerEnforced`-Einstellung als seinen Wert an. Mit anderen Worten, der IAM-Benutzer kann nur dann Buckets erstellen, wenn er die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft festgelegt und ACLs deaktiviert hat. Sie können diese Richtlinie auch als Grenz-SCP für Ihre AWS Organisation verwenden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RequireBucketOwnerFullControl",
            "Action": "s3:CreateBucket",
            "Effect": "Deny",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-object-ownership": "BucketOwnerEnforced"
                }
            }
```

```
        }
    ]
}
```

Erfordern der bucket-owner-full-control vordefinierten ACL für Amazon S3-**PUT**Operationen (Bucket-Eigentümer bevorzugt)

Mit der bevorzugten Einstellung des Bucket-Eigentümers für Object Ownership besitzen Sie als Bucket-Eigentümer die volle Kontrolle über neue Objekte, die andere Konten mit der vordefinierten `bucket-owner-full-control`-ACL in Ihren Bucket schreiben. Wenn jedoch andere Konten Objekte in Ihren Bucket schreiben, ohne die `bucket-owner-full-control` vordefinierte ACL behält der Objekt-Writer den vollen Zugriff auf die Kontrolle. Sie als Bucket-Eigentümer können eine Bucket-Richtlinie implementieren, die Schreibvorgänge nur zulässt, wenn sie die von `bucket-owner-full-control` vordefinierte ACL angeben.

> **Note**
>
> Wenn Sie ACLs mit der Einstellung „Bucket-Eigentümer erzwungen“ deaktiviert haben, besitzen Sie als Bucket-Eigentümer automatisch alle Objekte in Ihrem Bucket und haben die volle Kontrolle über diese Objekte. Sie müssen diesen Abschnitt nicht verwenden, um Ihre Bucket-Richtlinie zu aktualisieren, um den Objekteigentümer für den Bucket-Eigentümer durchzusetzen.

Die folgende Bucket-Richtlinie gibt an, dass das Konto *111122223333* nur dann Objekte in *DOC-EXAMPLE-BUCKET* hochladen kann, wenn die ACL des Objekts auf `bucket-owner-full-control` festgelegt ist. Achten Sie darauf, *111122223333* durch Ihr Konto und *DOC-EXAMPLE-BUCKET* durch den Namen Ihres Buckets zu ersetzen.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Only allow writes to my bucket with bucket owner full control",
         "Effect": "Allow",
         "Principal": {
            "AWS": [
               "arn:aws:iam::111122223333:user/ExampleUser"
            ]
```

```
         },
         "Action": [
            "s3:PutObject"
         ],
         "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
         "Condition": {
            "StringEquals": {
               "s3:x-amz-acl": "bucket-owner-full-control"
            }
         }
      }
   ]
}
```

Im Folgenden finden Sie ein Beispiel für eine Kopieroperation, die die vordefinierte `bucket-owner-full-control`-ACL über die AWS Command Line Interface (AWS CLI) einschließt.

```
aws s3 cp file.txt s3://DOC-EXAMPLE-BUCKET --acl bucket-owner-full-control
```

Nachdem die Bucket-Richtlinie in Kraft gesetzt wurde und der Client die vordefinierte `bucket-owner-full-control`-ACL nicht enthält, schlägt der Vorgang fehl und der Uploader erhält den folgenden Fehler:

Beim Aufrufen der PutObject Operation ist ein Fehler (AccessDenied) aufgetreten: Zugriff verweigert.

> **Note**
>
> Wenn Clients nach dem Hochladen Zugriff auf Objekte benötigen, müssen Sie dem hochladenden Konto zusätzliche Berechtigungen erteilen. Informationen zum Erteilen von Zugriff auf Ihre Ressourcen für Konten finden Sie unter Beispiel-Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon-S3-Ressourcen.

Fehlerbehebung

Wenn Sie die Einstellung „Von Bucket-Besitzer erzwungen“ für S3 Object Ownership anwenden, werden die Zugriffssteuerungslisten (ACLs) deaktiviert und Sie als Bucket-Besitzer besitzen automatisch alle Objekte in Ihrem Bucket. ACLs wirken sich nicht mehr auf Berechtigungen für die Objekte in Ihrem Bucket aus. Sie können Richtlinien verwenden, um Berechtigungen zu erteilen. Alle PUT-Anfragen von S3 müssen entweder die vordefinierte ACL `bucket-owner-full-control`

oder keine ACL angeben. Andernfalls schlagen diese Anfragen fehl. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Wenn eine ungültige ACL angegeben ist oder Bucket-ACL-Berechtigungen Zugriff außerhalb Ihres AWS-Konto gewähren, werden möglicherweise die folgenden Fehlerantworten angezeigt.

AccessControlListNotSupported

Nachdem Sie die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft angewendet haben, sind ACLs deaktiviert. Anforderungen zum Festlegen von ACLs oder Aktualisieren von ACLs schlagen mit einem `400` Fehler fehl und geben den AccessControlListNotSupported Fehlercode zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt. Anfragen zum Lesen von ACLs geben immer eine Antwort zurück, die die volle Kontrolle für den Bucket-Eigentümer anzeigt. In Ihren PUT-Vorgängen müssen Sie entweder ACLs für den Bucket-Eigentümer mit vollem Zugriff angeben oder keine ACL angeben. Andernfalls schlagen Ihre PUT-Operationen fehl.

Der folgende `put-object` AWS CLI Beispielbefehl enthält die `public-read` vordefinierte ACL.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key object-key-name --body doc-example-body --acl public-read
```

Wenn der Bucket die Einstellung „Bucket-Eigentümer erzwungen“ verwendet, um ACLs zu deaktivieren, schlägt dieser Vorgang fehl und der Uploader erhält die folgende Fehlermeldung:

Beim AccessControlListNotSupportedAufrufen der -PutObject Operation ist ein Fehler aufgetreten (): Der Bucket lässt keine ACLs zu

InvalidBucketAclWithObjectOwnership

Wenn Sie die Einstellung „Bucket-Eigentümer erzwungen“ zum Deaktivieren von ACLs anwenden möchten, darf Ihre Bucket-ACL nur dem Bucket-Eigentümer die volle Kontrolle geben. Ihre Bucket-ACL kann keinen Zugriff auf eine externe AWS-Konto oder eine andere Gruppe gewähren. Wenn Ihre `CreateBucket` Anforderung beispielsweise „Bucket-Eigentümer erzwungen“ festlegt und eine Bucket-ACL angibt, die Zugriff auf eine externe bietet AWS-Konto, schlägt Ihre Anforderung mit einem `400` Fehler fehl und gibt den InvalidBucketAclWithObjectOwnership Fehlercode zurück. Wenn Ihre `PutBucketOwnershipControls`-Anfrage die Einstellung Bucket-Eigentümer erzwungen mit einer Bucket-ACL festlegt, die anderen Berechtigungen erteilt, schlägt die Anfrage ebenfalls fehl.

Example : Vorhandene Bucket-ACL gewährt öffentlichen Lesezugriff

Wenn beispielsweise eine vorhandene Bucket-ACL öffentlichen Lesezugriff gewährt, können Sie die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft erst anwenden, wenn Sie diese ACL-Berechtigungen zu einer Bucket-Richtlinie migrieren und Ihre Bucket-ACL auf die private Standard-ACL zurücksetzen. Weitere Informationen finden Sie unter Voraussetzungen für die Deaktivierung von ACLs.

Diese Beispiel-Bucket-ACL gewährt öffentlichen Lesezugriff:

```
{
    "Owner": {
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

Der folgende `put-bucket-ownership-controls` AWS CLI Beispielbefehl wendet die Einstellung „Bucket-Eigentümer erzwungen“ für Object Ownership an:

```
aws s3api put-bucket-ownership-controls --bucket DOC-EXAMPLE-BUCKET --ownership-
controls Rules=[{ObjectOwnership=BucketOwnerEnforced}]
```

Da die Bucket-ACL öffentlichen Lesezugriff gewährt, schlägt die Anforderung fehl und gibt den folgenden Fehlercode zurück:

Beim Aufrufen der - PutBucketOwnershipControls Operation ist ein Fehler (InvalidBucketAclWithObjectOwnership) aufgetreten: Für den Bucket dürfen keine ACLs ObjectOwnershipmit der BucketOwnerEnforced Einstellung von festgelegt werden

Protokollierung und Überwachung in Amazon S3

Die Überwachung ist wichtig, um die Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon S3 und Ihrer - AWS Lösungen aufrechtzuerhalten. Sammeln Sie Überwachungsdaten aller Bestandteile Ihrer - AWS Lösung, damit Sie Ausfälle an mehreren Punkten leichter debuggen können. AWS bietet mehrere Tools zur Überwachung Ihrer Amazon S3-Ressourcen und zur Reaktion auf potenzielle Vorfälle.

Weitere Informationen finden Sie unter Überwachen von Amazon S3.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Amazon- CloudWatch Alarme

Mithilfe von Amazon- CloudWatch Alarmen überwachen Sie eine einzelne Metrik über einen von Ihnen angegebenen Zeitraum. Wenn die Metrik einen bestimmten Schwellenwert überschreitet, wird eine Benachrichtigung an ein Amazon SNS-Thema oder eine AWS Auto Scaling Richtlinie gesendet. CloudWatch Alarme rufen keine Aktionen auf, da sie sich in einem bestimmten Status befinden. Der Status muss sich stattdessen geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

AWS CloudTrail Protokolle

CloudTrail bietet eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem - AWS Service in Amazon S3 durchgeführten Aktionen. Anhand der von CloudTrailgesammelten Informationen können Sie die an Amazon S3 gestellte Anfrage, die IP-Adresse, von der die Anfrage gestellt wurde, den Initiator der Anfrage, den Zeitpunkt der Anfrage und zusätzliche Details bestimmen. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail.

Amazon-S3-Zugriffsprotokolle

Server-Zugriffsprotokolle enthalten detaillierte Aufzeichnungen über die Anfragen, die an einen Bucket gestellt wurden. Server-Zugriffsprotokolle sind für viele Anwendungen nützlich.

Beispielsweise können Zugriffsprotokoll-Informationen bei Sicherheits- und Zugriffsprüfungen nützlich sein. Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

AWS Trusted Advisor

Trusted Advisor stützt sich auf bewährte Methoden, die sich aus der Betreuung von Hunderttausenden von AWS Kunden ergeben haben. Trusted Advisor überprüft Ihre - AWS Umgebung und gibt dann Empfehlungen, wenn sich Möglichkeiten ergeben, Geld zu sparen, die Systemverfügbarkeit und -leistung zu verbessern oder Sicherheitslücken zu schließen. Alle - AWS Kunden haben Zugriff auf fünf Trusted Advisor Prüfungen. Kunden mit einem Business- oder Enterprise-Supportplan können alle Trusted Advisor Prüfungen anzeigen.

Trusted Advisor hat die folgenden AmazonS3-related Prüfungen:

- Protokollierungskonfiguration von Amazon-S3-Buckets.
- Sicherheitsprüfungen für Amazon-S3-Buckets mit offenen Zugriffsberechtigungen.
- Fehlertoleranzprüfungen für Amazon-S3-Buckets, für die keine Versionsverwaltung aktiviert oder deren Versionsverwaltung ausgesetzt ist

Weitere Informationen finden Sie unter AWS Trusted Advisor im AWS Support - Benutzerhandbuch.

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls die Protokollierung und Überwachung:

- Identify and audit all your Amazon S3 buckets
- Implement monitoring using Amazon Web Services monitoring tools
- Aktivieren AWS Config
- Enable Amazon S3 server access logging
- Use CloudTrail
- Monitor Amazon Web Services security advisories

Compliance-Validierung für Amazon S3

Die Sicherheit und Compliance von Amazon S3 wird von externen Prüfern im Rahmen verschiedener AWS -Compliance-Programme bewertet, darunter die folgenden:

- System and Organization Controls (SOC)
- Payment Card Industry Data Security Standard (PCI DSS)
- Federal Risk and Authorization Management Program (FedRAMP)
- Health Insurance Portability and Accountability Act (HIPAA)

AWS bietet eine häufig aktualisierte Liste der - AWS Services, die in den Geltungsbereich bestimmter Compliance-Programme fallen, unter AWS -Services im Geltungsbereich nach Compliance-Programm.

Auditberichte von Drittanbietern können Sie mit herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte in AWS Artifact herunterladen.

Weitere Informationen zu AWS -Compliance-Programmen finden Sie unter AWS -Compliance-Programme.

Welche Compliance-Verpflichtungen Sie bei der Verwendung von Amazon S3 haben, hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihrer Organisation und den geltenden Gesetzen und Vorschriften ab. Wenn Ihre Nutzung von Amazon S3 der Compliance von Standards wie HIPAA, PCI oder FedRAMP unterliegt, stellt AWS Ressourcen zur Unterstützung bereit:

- Kurzanleitungen für Sicherheit und Compliance, in denen Überlegungen und Schritte zur Architektur für die Bereitstellung von sicherheits- und Compliance-orientierten Basisumgebungen in erörtert werden AWS.
- Architekturerstellung für HIPAA-Sicherheit und -Compliance beschreibt, wie Unternehmen verwenden AWS , um sie bei der Erfüllung der HIPAA-Anforderungen zu unterstützen.
- AWS Compliance-Ressourcen bieten mehrere verschiedene Arbeitsmappen und Leitfäden, die für Ihre Branche und Ihren Standort möglicherweise relevant sind.
- AWS Config Mit können Sie bewerten, zu welchem Grad die Konfiguration Ihrer Ressourcen den internen Vorgehensweisen, Branchenrichtlinien und Vorschriften entspricht.
- AWS Security Hub bietet Ihnen einen umfassenden Überblick über Ihren Sicherheitsstatus in AWS und hilft Ihnen dabei, Ihre Compliance mit den Sicherheitsstandards und bewährten Methoden der Branche zu überprüfen.

- Verwenden der S3-Objektsperre unterstützt Sie dabei, technische Anforderungen von Regulierungsbehörden für Finanzdienstleistungen (wie SEC, FINRA und CFTC) einzuhalten, die den Datenspeicher „Write Once Read Many“ (WORM) für bestimmte Arten von Büchern und Datensatzinformationen erfordern.
- Amazon S3 Inventory können Sie bei der Prüfung und Meldung des Replikations- und Verschlüsselungsstatus Ihrer Objekte für Unternehmens-, Compliance- und regulatorische Anforderungen verwenden.

Ausfallsicherheit bei Amazon S3

Die AWS globale -Infrastruktur basiert auf -Regionen und Availability Zones. AWS-Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die mit einem Netzwerk mit geringer Latenz, hohem Durchsatz und hoher Redundanz verbunden sind. Diese Availability Zones bieten Ihnen eine effektive Methode zum Entwerfen und Betreiben von Anwendungen und Datenbanken. Availability Zones sind in noch größeren Ausmaß hochverfügbar und sie sind fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren. Wenn Sie Ihre Daten über größere geografische Entfernungen replizieren müssen, können Sie verwendenReplizieren von Objekten, was ein automatisches, asynchrones Kopieren von Objekten über Buckets hinweg in verschiedenen ermöglicht AWS-Regionen.

Jede AWS-Region hat mehrere Availability Zones. Sie können Ihre Anwendungen über mehrere Availability Zones in derselben Region bereitstellen, um eine bessere Fehlertoleranz und niedriger Latenz zu erzielen. Availability Zones sind mit schnellen, privaten Glasfasernetzwerken verbunden. Dies ermöglicht Ihnen die Nutzung von Anwendungen, für die ein automatischer, unterbrechungsfreier Failover zwischen den Availability Zones eingerichtet ist.

Weitere Informationen zu AWS-Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Neben der AWS globalen -Infrastruktur stellt Amazon S3 verschiedene Funktionen bereit, um Ihren Anforderungen an Ausfallsicherheit und Datensicherung gerecht zu werden.

Lebenszyklus-Konfiguration

Eine Lebenszyklus-Konfiguration besteht aus einer Reihe von Regeln, mit denen Aktionen definiert werden, die Amazon S3 auf eine Gruppe von Objekten anwendet. Mithilfe der Konfigurationsregeln für den Lebenszyklus können Sie Amazon S3 anweisen, Objekte in kostengünstigere Speicherklassen zu übergeben bzw. zu archivieren oder zu löschen. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Versioning

Das Versioning ermöglicht Ihnen, mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Sie können Versioning verwenden, um sämtliche Versionen aller Objekte in Ihrem Amazon S3 Bucket zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leicht wiederherstellen. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

S3-Objektsperre

Mit der S3-Objektsperre können Sie Objekte anhand des Modells Write Once Read Many (WORM) speichern. Mit der S3-Objektsperre können Sie für einen festen Zeitraum oder auf unbegrenzte Zeit verhindern, dass ein Objekt gelöscht oder überschrieben wird. Sie können die S3-Objektsperre verwenden, um regulatorische Anforderungen einzuhalten, die die WORM-Speicherung verlangen, oder um eine zusätzliche Schutzebene gegen Objektänderungen und -löschungen einzurichten. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

Speicherklassen

Amazon S3 bietet je nach den Anforderungen Ihrer Workload eine Reihe von Speicherklassen an. Die Speicherklassen S3 Standard-IA und S3 One Zone-IA sind für Daten konzipiert, auf die Sie etwa einmal im Monat zugreifen und auf Millisekunden zugreifen müssen. Die Speicherklasse S3 Glacier Instant Retrieval ist für langlebige Archivdaten konzipiert, auf die Sie mit Millisekunden-Zugriff zugreifen, auf den Sie etwa einmal pro Quartal zugreifen. Für Archivdaten, die keinen sofortigen Zugriff erfordern, wie zum Beispiel Backups, können Sie die Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive verwenden. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls die Ausfallsicherheit:

- Enable versioning
- Consider Amazon S3 cross-region replication
- Identify and audit all your Amazon S3 buckets

Verschlüsselung von Amazon-S3-Sicherungen

Wenn Sie Backups mit Amazon S3 speichern, hängt die Verschlüsselung Ihrer Backups von der Konfiguration dieser Buckets ab. Die Amazon-S3-Standard-Verschlüsselung bietet eine Methode zum Festlegen des Verhaltens der Standard-Verschlüsselung für einen S3-Bucket. Sie können die Standard-Verschlüsselung in einem Bucket festlegen, sodass alle Objekte beim Speichern im Bucket verschlüsselt werden. Die Standardverschlüsselung unterstützt Schlüssel, die in AWS KMS (SSE-KMS) gespeichert sind. Weitere Informationen finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.

Weitere Informationen zu Versioning und Objektsperre finden Sie in den folgenden Themen: Verwenden der Versioning in S3-Buckets Verwenden der S3-Objektsperre

Infrastruktursicherheit in Amazon S3

Als verwalteter Service ist Amazon S3 durch die AWS globalen Verfahren zur Gewährleistung der Netzwerksicherheit von geschützt, die in der Sicherheitssäule des AWS Well-Architected Framework beschrieben sind.

Der Zugriff auf Amazon S3 über das Netzwerk erfolgt über AWS veröffentlichte APIs. Clients müssen Transport Layer Security (TLS) 1.2 unterstützen. Wir empfehlen, auch TLS 1.3 zu unterstützen. (Weitere Informationen zu dieser Empfehlung finden Sie unter Schnellere AWS Cloud-Verbindungen mit TLS 1.3 im AWS -Sicherheitsblog.) Clients müssen außerdem Cipher Suites mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Darüber hinaus müssen Anforderungen mit AWS Signature V4 oder AWS Signature V2 signiert werden, sodass gültige Anmeldeinformationen bereitgestellt werden müssen.

Diese APIs lassen sich von einem beliebigen Netzwerkstandort aus aufrufen. Da Amazon S3 jedoch ressourcenbasierte Zugriffsrichtlinien unterstützt, kann es zu Einschränkungen bezüglich der Quell-IP-Adresse kommen. Sie können auch Amazon S3 Bucket-Richtlinien verwenden, um den Zugriff auf Buckets von bestimmten Virtual Private Cloud (VPC)-Endpunkten oder bestimmten VPCs aus zu steuern. Dadurch wird der Netzwerkzugriff auf einen bestimmten Amazon S3-Bucket effektiv nur von der spezifischen VPC innerhalb des AWS Netzwerks isoliert. Weitere Informationen finden Sie unter Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien.

Die folgenden bewährten Sicherheitsmethoden beziehen sich ebenfalls auf die Sicherheit der Infrastruktur von Amazon S3:

- Consider VPC endpoints for Amazon S3 access
- Identify and audit all your Amazon S3 buckets

Konfigurations- und Schwachstellenanalyse in Amazon S3

AWS kümmert sich um grundlegende Sicherheitsaufgaben wie Gastbetriebssystem (OS) und Datenbank-Patching, Firewall-Konfiguration und Notfallwiederherstellung. Diese Verfahren wurden von qualifizierten Dritten überprüft und zertifiziert. Weitere Informationen finden Sie in den folgenden Ressourcen:

- Compliance-Validierung für Amazon S3
- Modell der übergreifenden Verantwortlichkeit
- Amazon Web Services – Übersicht über Sicherheitsverfahren

Die folgenden bewährten Sicherheitsmethoden beinhalten auch die Adresskonfiguration und Schwachstellenanalyse in Amazon S3:

- Identify and audit all your Amazon S3 buckets
- Aktivieren AWS Config

Bewährte Methoden für die Sicherheit in Amazon S3

Amazon S3 enthält eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden stellen allgemeine Richtlinien und keine vollständige Sicherheitslösung dar. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Empfehlungen und nicht als bindend ansehen.

Themen

Bewährte Methoden für die Sicherheit in Amazon S3

Die folgenden bewährten Methoden für Amazon S3 können dazu beitragen, Sicherheitsvorfälle zu verhindern.

Deaktivieren von Zugriffssteuerungslisten (ACLs)

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, und ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Datenzugriff ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine Zugriffssteuerungslisten (ACLs) mehr. Wir empfehlen Ihnen, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Sie können die Einstellung „Bucket owner enforced“ (Bucket-Eigentümer erzwungen) für S3 Object Ownership anwenden, um ACLs zu deaktivieren und und das Eigentum an jedem Objekt in Ihrem Bucket zu übernehmen. Wenn Sie ACLs deaktivieren, können Sie einfach einen Bucket mit Objekten verwalten, die von verschiedenen AWS-Konten hochgeladen wurden.

Wenn ACLs deaktiviert sind, basiert die Zugriffskontrolle für Ihre Daten auf Richtlinien, wie die folgenden:

- AWS Identity and Access Management (IAM)-Benutzerrichtlinien

- S3-Bucket-Richtlinien
- Richtlinien für Virtual Private Cloud (VPC)-Endpunkte
- AWS Organizations Service-Kontrollrichtlinien (SCPs)

Durch Deaktivieren von ACLs werden Berechtigungsverwaltung und Auditing vereinfacht. ACLs sind für neu erstellte Buckets standardmäßig deaktiviert. Sie können ACLs auch für vorhandene Buckets deaktivieren. Wenn Sie einen vorhandenen Bucket haben, der bereits Objekte enthält, sind die Objekt- und Bucket-ACLs nach dem Deaktivieren von ACLs nicht mehr Teil des Zugriffsbewertungsprozesses. Stattdessen wird der Zugriff auf der Grundlage von Richtlinien gewährt oder verweigert.

Bevor Sie ACLs deaktivieren, müssen Sie folgende Schritte ausführen:

- Überprüfen Sie Ihre Bucket-Richtlinie, um sicherzustellen, dass sie alle Möglichkeiten abdeckt, wie Sie außerhalb Ihres Kontos Zugriff auf Ihren Bucket gewähren möchten.
- Setzen Sie Ihre Bucket-ACL auf die Standardeinstellung zurück (volle Kontrolle für den Bucket-Eigentümer).

Nachdem Sie ACLs deaktiviert haben, gilt Folgendes:

- Ihr Bucket akzeptiert nur PUT-Anforderungen, die keine ACL angeben, oder PUT-Anforderungen mit ACLs, bei denen der Bucket-Eigentümer die volle Kontrolle hat. Zu diesen ACLs gehören die vordefinierte ACL `bucket-owner-full-control` oder gleichwertige Formen dieser ACL, die in XML ausgedrückt sind.
- Bestehende Anwendungen, die ACLs mit voller Kontrolle des Bucket-Eigentümers unterstützen, sehen keine Auswirkungen
- PUT -Anforderungen, die andere ACLs enthalten (z. B. benutzerdefinierte Erteilungen für bestimmte AWS-Konten), schlagen fehl und geben einen HTTP-Statuscode `400 (Bad Request)` mit dem Fehlercode zurück`AccessControlListNotSupported`.

Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Achten Sie darauf, dass für Ihre Amazon-S3-Buckets die korrekten Richtlinien gelten und dass sie nicht öffentlich zugänglich sind.

Sofern es nicht unbedingt notwendig ist, dass jemand Ihren S3-Bucket im Internet lesen oder darin schreiben kann, stellen Sie sicher, dass der S3-Bucket nicht öffentlich ist. Im Folgenden werden einige Maßnahmen zum Blockieren des öffentlichen Zugriffs aufgeführt:

- Verwenden Sie S3 Block Public Access. Mit S3 Block Public Access können Sie problemlos zentrale Kontrollen zur Beschränkung des öffentlichen Zugriffs auf Ihre Amazon-S3-Ressourcen einrichten. Diese zentralen Kontrollen werden unabhängig davon durchgesetzt, wie die Ressourcen erstellt werden. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.
- Identifizieren Sie Amazon-S3-Bucket-Richtlinien, die eine Platzhalteridentität zulassen, z. B. `"Principal": "*"` (was praktisch „jeder“ bedeutet). Halten Sie auch nach Richtlinien Ausschau, die die Platzhalteraktion `"*"` zulassen (was einem Benutzer erlaubt, jede Aktion im Amazon-S3-Bucket durchzuführen).
- Suchen Sie in ähnlicher Weise nach Amazon S3-Bucket-Zugriffssteuerungslisten (ACLs), die „Alle“ oder „Alle authentifizierten AWS Benutzer“ Lese-, Schreib- oder Vollzugriffsberechtigungen bieten.
- Verwenden Sie die API-Operation `ListBuckets`, um alle Ihre Amazon-S3-Buckets zu scannen. Ermitteln Sie dann mit `GetBucketAcl`, `GetBucketWebsite` und `GetBucketPolicy`, ob der Bucket über richtlinienkonforme Zugriffskontrollen und Konfiguration verfügt.
- Verwenden Sie AWS Trusted Advisor diese Option, um Ihre Amazon-S3-Implementierung zu überprüfen.
- Erwägen Sie die Implementierung fortlaufender Erkennungskontrollen mithilfe der verwalteten AWS-Config-Regelns3-bucket-public-read-prohibited und s3-bucket-public-write-prohibited.

Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3.

Implementieren des Zugriffs mit geringsten Berechtigungen

Bei der Vergabe von Berechtigungen entscheiden Sie, wer welche Berechtigungen für welche Amazon-S3-Ressourcen erhält. Sie aktivieren die spezifischen Aktionen, die daraufhin für die betreffenden Ressourcen erlaubt sein sollen. Aus diesem Grund sollten Sie nur Berechtigungen gewähren, die zum Ausführen einer Aufgabe erforderlich sind. Die Implementierung der geringstmöglichen Zugriffsrechte ist eine grundlegende Voraussetzung zum Reduzieren des Sicherheitsrisikos und der Auswirkungen, die aufgrund von Fehlern oder böswilligen Absichten entstehen könnten.

Die folgenden Tools stehen zur Implementierung der geringstmöglichen Zugriffsrechte zur Verfügung:

- Amazon S3-Richtlinienaktionen und Berechtigungsgrenzen für IAM-Entitäten
- Bucket-Richtlinien und Benutzerrichtlinien

- Zugriffskontrolllisten (ACL) – Übersicht
- Service-Kontrollrichtlinien

Hilfreiche Informationen zur Entscheidung, welche der vorherigen Mechanismen Sie auswählen sollten, finden Sie unter Richtlinien für Zugriffsrichtlinien.

Verwenden von IAM-Rollen für Anwendungen und AWS-Services , die Amazon S3-Zugriff erfordern

Damit Anwendungen AWS-Services , die auf Amazon EC2 oder anderen ausgeführt werden, auf Amazon S3-Ressourcen zugreifen können, müssen sie in ihren AWS API-Anforderungen gültige AWS Anmeldeinformationen enthalten. Wir empfehlen, AWS Anmeldeinformationen nicht direkt in der Anwendung oder Amazon EC2Instance zu speichern. Dabei handelt es sich um langfristige Anmeldeinformationen, die nicht automatisch rotiert werden und bedeutende geschäftliche Auswirkungen haben könnten, wenn sie kompromittiert werden.

Verwalten Sie stattdessen mithilfe einer IAM-Rolle temporäre Anmeldeinformationen für Anwendungen und Services, die Zugriff auf Amazon S3 benötigen. Wenn Sie eine Rolle verwenden, müssen Sie keine langfristigen Anmeldeinformationen (wie Benutzername und Passwort oder Zugriffsschlüssel) an eine Amazon EC2 oder wie verteilen AWS-Service AWS Lambda. Die Rolle stellt temporäre Berechtigungen bereit, die Anwendungen verwenden können, wenn sie Aufrufe an andere - AWS Ressourcen tätigen.

Weitere Informationen finden Sie unter folgenden Themen im IAM-Benutzerhandbuch:

- IAM-Rollen
- Gängige Szenarien für Rollen: Benutzer, Anwendungen und Services

Erwägen der Verschlüsselung von Data-at-Rest

Es gibt die folgenden Optionen, Daten im Ruhezustand in Amazon S3 zu schützen:

- Serverseitige Verschlüsselung – Für alle Amazon S3-Buckets ist die Verschlüsselung standardmäßig konfiguriert, und alle neuen Objekte, die in einen S3-Bucket hochgeladen werden, werden im Ruhezustand automatisch verschlüsselt. Die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) ist die Standardverschlüsselungskonfiguration für jeden Bucket in Amazon S3. Um einen anderen Verschlüsselungstyp zu verwenden, können Sie entweder die Art der serverseitigen Verschlüsselung angeben, die in Ihren S3-PUT-Anfragen verwendet werden soll, oder Sie können die Standardverschlüsselungskonfiguration im Ziel-Bucket festlegen.

  Amazon S3 bietet auch die folgenden serverseitigen Verschlüsselungsoptionen:

- Serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS)
- Serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS)
- Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Weitere Informationen finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung.

- Clientseitige Verschlüsselung – Sie können Daten clientseitig verschlüsseln und die verschlüsselten Daten auf Amazon S3 hochladen. In diesem Fall verwalten Sie den Verschlüsselungsprozess, die Verschlüsselungsschlüssel und die zugehörigen Tools. Genau wie die serverseitige Verschlüsselung kann die client-seitige Verschlüsselung das Risiko reduzieren, indem die Daten mit einem Schlüssel verschlüsselt werden, der durch einen anderen Mechanismus gespeichert wird als der, der die Daten selbst speichert

  Amazon S3 bietet mehrere clientseitige Verschlüsselungsoptionen. Weitere Informationen finden Sie unter Schützen von Daten mithilfe der clientseitigen Verschlüsselung.

Erzwingen der Verschlüsselung von Daten während der Übertragung

Sie können HTTPS (TLS) verwenden, um zu verhindern, dass potenzielle Angreifer den Netzwerkverkehr abhören oder manipulieren, indem Sie person-in-the-middle - oder ähnliche Angriffe verwenden. Es wird empfohlen, nur verschlüsselte Verbindungen über HTTPS (TLS) unter Verwendung der Bedingung aws:SecureTransport in Ihren Amazon-S3-Bucket-Richtlinien zu erlauben.

Erwägen Sie außerdem die Implementierung fortlaufender Erkennungskontrollen mithilfe der verwalteten AWS Config -Regel s3-bucket-ssl-requests-only.

Erwägen der S3-Objektsperre

Mit der S3-Objektsperre können Sie Objekte anhand des Modells „Write Once Read Many“ (WORM) speichern. Die S3-Objektsperre kann unbeabsichtigtes oder unsachgemäßes Löschen von Daten verhindern. Sie können beispielsweise die S3-Objektsperre verwenden, um Ihre AWS CloudTrail Protokolle zu schützen.

Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

Aktivieren der S3-Versionsverwaltung

Die S3-Versionsverwaltung ermöglicht Ihnen, mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Sie können Versioning verwenden, um sämtliche Versionen aller

Objekte in Ihrem Bucket zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leicht wiederherstellen.

Erwägen Sie außerdem die Implementierung fortlaufender Erkennungskontrollen mithilfe der verwalteten AWS Config -Regel s3-bucket-versioning-enabled.

Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Erwägen einer regionsübergreifenden S3-Replikation

Auch wenn Amazon S3 Ihre Daten standardmäßig in mehreren entfernten Availability Zones speichert, machen es die Compliance-Anforderungen möglicherweise erforderlich, Daten in noch größeren Entfernungen zu speichern. Mit der regionsübergreifenden Replikation (CRR) von S3 können Sie Daten zwischen entfernten replizieren AWS-Regionen , um diese Anforderungen zu erfüllen. CRR ermöglicht automatisches, asynchrones Kopieren von Objekten über Buckets hinweg in verschiedenen AWS-Regionen. Weitere Informationen finden Sie unter Replizieren von Objekten.

> **Note**
>
> CRR erfordert, dass die Versionsverwaltung in Quell- und Ziel-S3-Buckets aktiviert ist.

Erwägen Sie außerdem die Implementierung fortlaufender Erkennungskontrollen mithilfe der verwalteten AWS Config -Regel s3-bucket-replication-enabled.

Erwägen von VPC-Endpunkten für den Amazon-S3-Zugriff

Ein Virtual Private Cloud (VPC)-Endpunkt für Amazon S3 ist eine logische Einheit innerhalb einer VPC, die nur Konnektivität mit Amazon S3 ermöglicht. VPC-Endpunkte können dazu beitragen, zu verhindern, dass Datenverkehr durch das offene Internet gelangt.

VPC-Endpunkte für Amazon S3 bieten mehrere Möglichkeiten, den Zugriff auf Ihre Amazon-S3-Daten zu steuern:

- Mithilfe von S3-Bucket-Richtlinien können Sie steuern, welche Anfragen, Benutzer oder Gruppen durch einen spezifischen VPC-Endpunkt erlaubt sind.
- Mit S3 Bucket-Richtlinien können Sie steuern, welche VPCs oder VPC-Endpunkte Zugriff auf Ihre S3-Buckets haben.

- Sie können eine Daten-Exfiltration verhindern, indem Sie eine VPC verwenden, die kein Internet-Gateway hat.

  Weitere Informationen finden Sie unter Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien.

Verwenden von verwalteten AWS Sicherheitsservices zur Überwachung der Datensicherheit

Mehrere verwaltete AWS Sicherheitsservices können Ihnen helfen, Sicherheits- und Compliance-Risiken für Ihre Amazon S3-Daten zu identifizieren, zu bewerten und zu überwachen. Diese Services können Ihnen auch dabei helfen, Ihre Daten vor diesen Risiken zu schützen. Zu diesen Services gehören automatisierte Erkennungs-, Überwachungs- und Schutzfunktionen, die darauf ausgelegt sind, von Amazon S3-Ressourcen für einen einzelnen AWS-Konto auf Ressourcen für Organisationen mit Tausenden von Konten zu skalieren.

Weitere Informationen finden Sie unter Überwachung der Datensicherheit mit verwalteten - AWS Sicherheitsservices.

Bewährte Methoden zur Überwachung und Prüfung von Amazon S3

Die folgenden bewährten Methoden für Amazon S3 können dabei helfen, potenzielle Sicherheitsschwächen und Vorfälle zu erkennen.

Identifizieren und Prüfen aller Ihrer Amazon-S3-Buckets

Die Identifikation Ihrer IT-Assets ist ein wichtiger Aspekt von Governance und Sicherheit. Es ist erforderlich, dass Sie alle Ihre Amazon-S3-Ressourcen überblicken, um ihren Sicherheitsstatus beurteilen und Maßnahmen gegen potenzielle Schwachstellen ergreifen zu können. Zum Prüfen Ihrer Ressourcen empfehlen wir Folgendes:

- Verwenden Sie den Tag-Editor, um sicherheits- und prüfungsrelevante Ressourcen zu identifizieren und zu markieren. Nutzen Sie dann diese Markierungen für die Suche nach den entsprechenden Ressourcen. Weitere Informationen finden Sie unter Suchen nach zu markierenden Ressourcen im Benutzerhandbuch zum Markieren von AWS Ressourcen.
- Verwenden Sie S3 Inventory für die Prüfung und Meldung des Replikations- und Verschlüsselungsstatus Ihrer Objekte für Unternehmens-, Compliance- und regulatorische Anforderungen. Weitere Informationen finden Sie unter Amazon S3 Inventory.
- Erstellen Sie Ressourcengruppen für Ihre Amazon-S3-Ressourcen. Weitere Informationen finden Sie unter Was sind Ressourcengruppen? im AWS Resource Groups -Benutzerhandbuch.

Implementieren der Überwachung mithilfe von AWS Überwachungstools

Die Überwachung ist wichtig, um die Zuverlässigkeit, Sicherheit, Verfügbarkeit und Leistung von Amazon S3 und Ihren - AWS Lösungen aufrechtzuerhalten. AWS bietet mehrere Tools und Services, die Sie bei der Überwachung von Amazon S3 und Ihren anderen unterstützen AWS-Services. Sie können beispielsweise Amazon- CloudWatch Metriken für Amazon S3 überwachen, insbesondere die `DeleteRequests` Metriken `PutRequests`, `4xxErrors`, und `GetRequests`. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch und Überwachen von Amazon S3.

Ein zweites Beispiel finden Sie unter Beispiel: Amazon-S3-Bucket-Aktivität. In diesem Beispiel wird beschrieben, wie Sie einen CloudWatch Alarm erstellen, der ausgelöst wird, wenn ein Amazon S3-API-Aufruf an `PUT` oder `DELETE` eine Bucket-Richtlinie, einen Bucket-Lebenszyklus oder eine Bucket-Replikationskonfiguration oder an `PUT` eine Bucket-ACL erfolgt.

Aktivieren Sie die Amazon-S3-Server-Zugriffsprotokollierung

Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anfragen, die an einen Bucket gestellt wurden. Server-Zugriffsprotokolle können Sie bei Sicherheits- und Zugriffsprüfungen unterstützen; sie geben Informationen über Ihren Kundenstamm und helfen beim Verständnis der Amazon-S3-Rechnung. Weitere Informationen zur Aktivierung der Server-Zugriffsprotokollierung finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

Erwägen Sie auch die Implementierung kontinuierlicher detektivischer Kontrollen mithilfe der -s3-bucket-logging-enabled AWS Config verwalteten Regel.

Verwenden Sie AWS CloudTrail

AWS CloudTrail bietet eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem AWS-Service in Amazon S3 durchgeführten Aktionen. Sie können die von gesammelten Informationen verwenden CloudTrail , um Folgendes zu bestimmen:

- Die Anforderung, die an Amazon S3 gestellt wurde
- Die IP-Adresse, von der die Anforderung erfolgt ist
- Wer die Anforderung gestellt hat
- Wann die Anforderung gestellt wurde
- Zusätzliche Details zur Anforderung

Sie können beispielsweise CloudTrail Einträge für PUT Aktionen identifizieren, die sich auf den Datenzugriff auswirken, insbesondere `PutBucketAcl`, `PutObjectAclPutBucketPolicy`, und `PutBucketWebsite`.

Wenn Sie Ihr einrichten AWS-Konto, CloudTrail ist standardmäßig aktiviert. Sie können aktuelle Ereignisse in der - CloudTrail Konsole anzeigen. Um eine fortlaufende Aufzeichnung der Aktivitäten und Ereignisse für Ihre Amazon S3-Buckets zu erstellen, können Sie einen Trail in der - CloudTrail Konsole erstellen. Weitere Informationen finden Sie unter Protokollieren von Datenereignissen im Benutzerhandbuch für AWS CloudTrail .

Wenn Sie einen Trail erstellen, können Sie so konfigurieren, CloudTrail dass Datenereignisse protokolliert werden. Datenereignisse sind Datensätze von Ressourcen-Vorgänge, die auf oder innerhalb einer Ressource ausgeführt werden. In Amazon S3 zeichnen Datenereignisse API-Aktivitäten auf Objektebene für einzelne Buckets auf. CloudTrail unterstützt eine Teilmenge von Amazon S3-API-Operationen auf Objektebene, z. B. `GetObjectDeleteObject`, und `PutObject`. Weitere Informationen zur CloudTrail Funktionsweise von mit Amazon S3 finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail. In der Amazon-S3-Konsole können Sie Ihre S3-Buckets auch dafür konfigurieren, Aktivieren der CloudTrail Ereignisprotokollierung für S3-Buckets und -Objekte.

AWS Config stellt eine verwaltete Regel (`cloudtrail-s3-dataevents-enabled`) bereit, mit der Sie bestätigen können, dass mindestens ein CloudTrail Trail Datenereignisse für Ihre S3-Buckets protokolliert. Weitere Informationen finden Sie unter cloudtrail-s3-dataevents-enabled im AWS Config -Entwicklerhandbuch.

Aktivieren AWS Config

Einige der in diesem Thema aufgeführten bewährten Methoden schlagen vor, rules AWS Config . AWS Config helft Ihnen, die Konfigurationen Ihrer Ressourcenkonfigurationen AWS resources. AWS Config monitors zu bewerten, zu prüfen und zu bewerten, sodass Sie die aufgezeichneten Konfigurationen anhand der gewünschten sicheren Konfigurationen bewerten können. Mit können AWS Config Sie Folgendes tun:

- Prüfen von Änderungen der Konfigurationen und Beziehungen zwischen AWS -Ressourcen
- Untersuchen detaillierter Ressourcenkonfigurationsverläufe
- Bestimmen der allgemeinen Compliance im Hinblick auf die Konfigurationen, die in Ihren internen Leitlinien angegeben sind

Die Verwendung von AWS Config kann Ihnen helfen, die Compliance-Prüfung, Sicherheitsanalyse, Änderungsmanagement und betriebliche Fehlerbehebung zu vereinfachen. Weitere Informationen finden Sie unter Einrichten von AWS Config mit der Konsole im AWS Config Entwicklerhandbuch für . Wenn Sie die aufzuzeichnenden Ressourcentypen angeben, stellen Sie sicher, dass Sie Amazon-S3-Ressourcen mit einbeziehen.

> **Important**
>
> AWS Config -verwaltete Regeln unterstützen bei der Auswertung von Amazon S3-Ressourcen nur Allzweck-Buckets. AWS Config zeichnet keine Konfigurationsänderungen für Verzeichnis-Buckets auf. Weitere Informationen finden Sie unter AWS Config Verwaltete -Regeln und Liste der AWS Config verwalteten -Regeln im AWS Config -Entwicklerhandbuch.

Ein Beispiel für die Verwendung von finden Sie unter Verwendung von zur Überwachung von und Reaktion auf Amazon-S3-Buckets AWS Config, die öffentlichen Zugriff zulassen im AWS -Sicherheitsblog. AWS Config Amazon S3

Entdecken von sensiblen Daten mithilfe von Amazon Macie

Amazon Macie ist ein Sicherheitsservice, der sensible Daten mithilfe von Machine Learning und Musterabgleich erkennt. Macie bietet Einblick in Datensicherheitsrisiken und ermöglicht automatischen Schutz vor diesen Risiken. Mit Macie können Sie die Erkennung und Meldung sensibler Daten in Ihrem Amazon-S3-Datenbestand automatisieren, um die Daten besser zu verstehen, die Ihre Organisation in S3 speichert.

Um sensible Daten mit Macie zu erkennen, können Sie integrierte Kriterien und Techniken verwenden, die darauf ausgelegt sind, eine lange und wachsende Liste vertraulicher Datentypen für viele Länder und Regionen zu erkennen. Diese sensiblen Datentypen umfassen mehrere Arten von persönlich identifizierbaren Informationen (PII), Finanzdaten und Anmeldeinformationen. Sie können auch benutzerdefinierte Kriterien verwenden, die Sie definieren – reguläre Ausdrücke, die Textmuster zum Abgleichen definieren, und optional Zeichenfolgen und Näherungsregeln, um die Ergebnisse zu verfeinern.

Wenn Macie sensible Daten in einem S3-Objekt entdeckt, generiert es eine Sicherheitserkenntnis, um Sie zu benachrichtigen. Diese Erkenntnis enthält Informationen über das betroffene Objekt, die Art und Häufigkeit der Vorkommen der sensiblen Daten, die Macie gefunden hat, sowie

zusätzliche Details, die Ihnen bei der Untersuchung des betroffenen S3-Buckets und -Objekts helfen. Weitere Informationen finden Sie im Benutzerhandbuch für Amazon Macie.

Verwenden von S3 Storage Lens

S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. S3 Storage Lens analysiert Metriken, um kontextbezogene Empfehlungen zur Optimierung der Speicherkosten und zur Anwendung bewährter Datenschutzmethoden zu geben.

Mit S3 Storage Lens können Sie Metriken verwenden, um zusammenfassende Erkenntnisse zu gewinnen und z. B. herauszufinden, wie viel Speicher Sie in Ihrer gesamten Organisation haben oder welche Buckets und Präfixe am schnellsten wachsen. Außerdem können Sie anhand der Metriken von S3 Storage Lens umfassende Möglichkeiten zur Kostenoptimierung aufdecken, bewährte Methoden für den Datenschutz und die Zugriffsverwaltung implementieren und die Leistung von Anwendungs-Workloads verbessern.

Sie können beispielsweise Buckets identifizieren, für die keine S3-Lebenszyklusregeln gelten, damit unvollständige mehrteilige Uploads, die älter als 7 Tage sind, abgebrochen werden. Sie können auch Buckets identifizieren, die nicht den bewährten Datenschutzmethoden entsprechen, z. B. die Verwendung von S3 Replication oder S3 Versionierung. Weitere Informationen finden Sie unter Grundlegendes zu Amazon S3 Storage Lens.

Überwachen von AWS -Sicherheitsempfehlungen

Wir raten Ihnen, die in Trusted Advisor für Ihr AWS-Konto geposteten Sicherheitsempfehlungen regelmäßig zu überprüfen. Achten Sie dabei besonders auf Warnungen zu Amazon-S3-Buckets mit „offenen Zugriffsberechtigungen“. Sie können diesen Schritt programmgesteuert durchführen, indem Sie describe-trusted-advisor-checks verwenden.

Überwachen Sie außerdem aktiv die primäre E-Mail-Adresse, die bei jedem Ihrer registriert ist AWS-Konten. AWS verwendet diese E-Mail-Adresse, um Sie bei auftretenden Sicherheitsproblemen zu kontaktieren, die Sie betreffen könnten.

AWS Operative Probleme mit großen Auswirkungen werden im AWS Health Dashboard - Servicezustand gepostet. Operative Probleme werden über AWS Health Dashboard auch in den einzelnen Konten veröffentlicht. Weitere Informationen finden Sie in der AWS Health - Dokumentation.

Überwachung der Datensicherheit mit verwalteten - AWS Sicherheitsservices

Mehrere verwaltete AWS Sicherheitsservices können Ihnen helfen, Sicherheits- und Compliance-Risiken für Ihre Amazon S3-Daten zu identifizieren, zu bewerten und zu überwachen. Sie können Sie auch dabei unterstützen, Ihre Daten vor diesen Risiken zu schützen. Zu diesen Services gehören automatisierte Erkennungs-, Überwachungs- und Schutzfunktionen, die darauf ausgelegt sind, von Amazon S3-Ressourcen für einen einzelnen AWS-Konto auf Ressourcen für Organisationen mit Tausenden von zu skalieren AWS-Konten.

AWS -Erkennungs- und -Antwortservices können Ihnen helfen, potenzielle Sicherheitsfehlkonfigurationen, Bedrohungen oder unerwartetes Verhalten zu identifizieren, sodass Sie schnell auf potenziell unbefugte oder böswillige Aktivitäten in Ihrer -Umgebung reagieren können. - AWS Datenschutzservices können Ihnen helfen, Ihre Daten, Konten und Workloads zu überwachen und vor unbefugtem Zugriff zu schützen. Sie können Ihnen auch dabei helfen, sensible Daten, wie persönlich identifizierbare Informationen (PII), in Ihrem Amazon-S3-Datenbestand zu finden.

Um Sie bei der Identifizierung und Bewertung von Datensicherheits- und Compliance-Risiken zu unterstützen, generieren verwaltete AWS -Sicherheitsservices Erkenntnisse, um Sie über potenzielle Sicherheitsereignisse oder -probleme mit Ihren Amazon-S3-Daten zu informieren. Die Erkenntnisse enthalten relevante Informationen, anhand derer Sie diese Risiken untersuchen, bewerten und entsprechend Ihren Vorfallreaktions-Workflows und -richtlinien handeln können. Über die einzelnen Services können Sie direkt auf die Erkenntnisdaten zugreifen. Sie können die Daten auch an andere Anwendungen, Services und Systeme senden, z. B. an Ihr Sicherheitsvorfall- und Ereignismanagementsystem (SIEM).

Um die Sicherheit Ihrer Amazon S3-Daten zu überwachen, sollten Sie die Verwendung dieser verwalteten AWS Sicherheitsservices in Betracht ziehen.

Amazon GuardDuty

Amazon GuardDuty ist ein Service zur Erkennung von Bedrohungen, der Ihre - AWS-Konten und -Workloads kontinuierlich auf böswillige Aktivitäten überwacht und detaillierte Sicherheitserkenntnisse für Transparenz und Abhilfe liefert.

Mit der S3-Schutzfunktion in können Sie so konfigurieren GuardDuty, GuardDuty dass AWS CloudTrail Verwaltungs- und Datenereignisse für Ihre Amazon S3-Ressourcen analysiert werden. GuardDuty überwacht diese Ereignisse auf böswillige und verdächtige Aktivitäten. Um die

Analyse zu unterstützen und potenzielle Sicherheitsrisiken zu identifizieren, GuardDuty verwendet Bedrohungsinformationen und Machine Learning.

GuardDuty kann verschiedene Arten von Aktivitäten für Ihre Amazon S3-Ressourcen überwachen. CloudTrail Verwaltungsereignisse für Amazon S3 umfassen beispielsweise Operationen auf Bucket-Ebene, wie `ListBuckets`, `DeleteBucket`und `PutBucketReplication`. CloudTrail Datenereignisse für Amazon S3 beinhalten Operationen auf Objektebene, wie `GetObject`, `ListObjects`und `PutObject`. Wenn ungewöhnliche oder potenziell böswillige Aktivitäten GuardDuty erkennt, generiert es eine Erkenntnis, die Sie benachrichtigt.

Weitere Informationen finden Sie unter Amazon S3 Protection in Amazon GuardDuty im Amazon-GuardDuty Benutzerhandbuch.

Amazon Detective

Amazon Detective vereinfacht den Ermittlungsprozess und hilft Ihnen, schnellere und effektivere Sicherheitsuntersuchungen durchzuführen. Detective bietet vordefinierte Datenaggregationen, Zusammenfassungen und Kontexte, mit denen Sie Art und Ausmaß möglicher Sicherheitsprobleme analysieren und bewerten können.

Detective extrahiert automatisch zeitbasierte Ereignisse, z. B. API-Aufrufe von AWS CloudTrail und Amazon VPC Flow Logs für Ihre AWS Ressourcen. Außerdem werden von Amazon generierte Erkenntnisse erfasst GuardDuty. Detective verwendet dann Machine Learning, statistische Analysen und die Graphentheorie, um Visualisierungen zu erstellen, mit denen Sie effektive Sicherheitsuntersuchungen schneller durchführen können.

Diese Visualisierungen bieten eine einheitliche, interaktive Ansicht des Ressourcenverhaltens und der Interaktionen zwischen ihnen im Zeitverlauf. Sie können dieses Verhaltensdiagramm untersuchen, um potenziell bösartige Aktionen wie fehlgeschlagene Anmeldeversuche oder verdächtige API-Aufrufe zu untersuchen. Sie können auch sehen, wie sich diese Aktionen auf Ressourcen wie S3-Buckets und -Objekte auswirken.

Weitere Informationen finden Sie im Administratorhandbuch für Amazon Detective.

IAM Access Analyzer

AWS Identity and Access Management Access Analyzer (IAM Access Analyzer) kann Ihnen helfen, Ressourcen zu identifizieren, die mit einer externen Entität geteilt werden. Sie können IAM Access Analyzer auch verwenden, um IAM-Richtlinien anhand der Richtliniengrammatik und bewährter Methoden zu validieren und IAM-Richtlinien basierend auf Zugriffsaktivitäten in Ihren AWS CloudTrail Protokollen zu generieren.

IAM Access Analyzer verwendet logisches Denken, um Ressourcenrichtlinien in Ihrer AWS Umgebung zu analysieren, z. B. Bucket-Richtlinien. Mit IAM Access Analyzer für S3 werden Sie benachrichtigt, wenn ein S3-Bucket so konfiguriert ist, dass jedem im Internet oder anderen , einschließlich Konten außerhalb Ihrer Organisation AWS-Konten, Zugriff gewährt wird. Beispielsweise könnte IAM Access Analyzer für S3 zeigen, dass ein Bucket über Lese- oder Schreibzugriff verfügt, der über eine Bucket-Zugriffssteuerungsliste (ACL), eine Bucket-Richtlinie, eine Richtlinie für Multi-Region Access Points oder eine Zugriffspunktrichtlinie bereitgestellt wird. Für jeden öffentlichen oder freigegebenen Bucket erhalten Sie Erkenntnisse, die die Quelle und die Ebene des öffentlichen oder freigegebenen Zugriffs angeben. Anhand dieser Erkenntnisse können Sie sofortige und präzise Korrekturmaßnahmen ergreifen, um den Bucket-Zugriff wie beabsichtigt wiederherzustellen.

Weitere Informationen finden Sie unter Überprüfen des Bucket-Zugriffs mit IAM Access Analyzer für S3.

Amazon Macie

Amazon Macie ist ein Datensicherheitsservice, der sensible Daten mithilfe von Machine Learning und Musterabgleich entdeckt, Einblicke in Datensicherheitsrisiken bietet und automatischen Schutz vor diesen Risiken ermöglicht.

Mit Macie können Sie die Erkennung und Meldung sensibler Daten in Ihren S3-Buckets automatisieren, um die Daten, die Ihre Organisation in Amazon S3 speichert, besser zu verstehen. Um sensible Daten zu erkennen, können Sie die von Macie bereitgestellten integrierten Kriterien und Techniken, benutzerdefinierte Kriterien, die Sie definieren, oder eine Kombination aus beiden verwenden. Wenn Macie sensible Daten in einem S3-Objekt entdeckt, generiert es eine Erkenntnis, um Sie zu benachrichtigen. Diese Erkenntnis enthält Informationen über den betroffenen Bucket und das entsprechende Objekt, die Art und Häufigkeit der Vorkommen der sensiblen Daten, die Macie gefunden hat, sowie zusätzliche Details, die Ihnen die Untersuchung erleichtern.

Macie stellt außerdem Statistiken und andere Daten zur Verfügung, die einen Einblick in die Sicherheitslage Ihrer Amazon-S3-Daten geben, wertet Ihre S3-Buckets automatisch aus und überwacht sie, um die Sicherheit und den Zugriff zu kontrollieren. Wenn Macie ein potenzielles Problem mit der Sicherheit oder dem Datenschutz erkennt, wie einen Bucket, der öffentlich zugänglich wird, generiert Macie eine Erkenntnis, die Sie überprüfen und bei Bedarf korrigieren können.

Weitere Informationen finden Sie im Benutzerhandbuch für Amazon Macie.

AWS Security Hub

AWS Security Hub ist ein Service zur Verwaltung des Sicherheitsstatus, der bewährte Sicherheitsprüfungen durchführt, Warnungen und Erkenntnisse aus mehreren Quellen in einem einzigen Format zusammenfasst und automatische Abhilfemaßnahmen ermöglicht.

Security Hub sammelt und stellt Daten zu Sicherheitserkenntnissen aus integrierten AWS Partner Network Sicherheitslösungen und bereit AWS-Services, einschließlich Amazon Detective, Amazon GuardDuty, IAM Access Analyzer und Amazon Macie . Es generiert auch eigene Erkenntnisse, indem kontinuierliche, automatisierte Sicherheitsprüfungen auf der Grundlage AWS von bewährten Methoden und unterstützten Industriestandards durchgeführt werden.

Security Hub korreliert und konsolidiert die Erkenntnisse dann anbieterübergreifend, um Ihnen dabei zu helfen, die wichtigsten Erkenntnisse zu priorisieren und zu verarbeiten. Es bietet auch Unterstützung für benutzerdefinierte Aktionen, mit denen Sie Antworten oder Abhilfemaßnahmen für bestimmte Klassen von Erkenntnissen aufrufen können.

Mit Security Hub können Sie den Sicherheits- und Compliance-Status Ihrer Amazon S3-Ressourcen bewerten. Dies ist Teil einer umfassenderen Analyse der Sicherheitslage Ihrer Organisation in einzelnen AWS-Regionen und über mehrere Regionen hinweg möglich. Dazu gehören die Analyse von Sicherheitstrends und die Identifizierung der Sicherheitsprobleme mit der höchsten Priorität. Sie können auch Erkenntnisse aus mehreren AWS-Regionen zusammenfassen und aggregierte Erkenntnisdaten aus einer einzelnen Region überwachen und verarbeiten.

Weitere Informationen finden Sie unter Kontrollen von Amazon Simple Storage Service im Benutzerhandbuch für AWS Security Hub .

Verwalten Ihres Amazon-S3-Speichers

Nachdem Sie Buckets erstellt und Objekte in Amazon S3 hochgeladen haben, können Sie Ihren Objektspeicher mithilfe von Funktionen wie Versioning, Speicherklassen, Objektsperre, Batch-Vorgänge, Replikation, Markierungen und mehr verwalten. Die folgenden Abschnitte enthalten detaillierte Informationen zu den Speicherverwaltungsfunktionen und Funktionen, die in Amazon S3 verfügbar sind.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Verwenden der Versioning in S3-Buckets

Das Versioning in Amazon S3 ermöglicht Ihnen, mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Sie können die S3-Versioning-Funktion verwenden, um sämtliche Versionen aller Objekte in Ihren Buckets zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leichter wiederherstellen. Wenn Sie das Versioning für einen Bucket aktivieren und Amazon S3 mehrere Schreibanforderungen für dasselbe Objekt gleichzeitig empfängt, werden alle Objekte gespeichert.

Versioning-fähige Buckets erlauben Ihnen, Objekte nach einem versehentlichen Löschen oder Überschreiben wiederherzustellen. Wenn Sie beispielsweise ein Objekt löschen, fügt Amazon S3 eine Löschmarkierung ein, anstatt das Objekt dauerhaft zu entfernen. Die Löschmarkierung wird zur aktuellen Objektversion. Wenn Sie ein Objekt überschreiben, entsteht eine neue Objektversion im Bucket. Sie können die vorherige Version immer wieder herstellen. Weitere Informationen finden Sie unter Löschen von Objekten aus einem versioning-fähigen Bucket.

Standardmäßig ist das S3-Versioning in Buckets deaktiviert und Sie müssen es explizit aktivieren. Weitere Informationen finden Sie unter Aktivieren des Versioning für Buckets.

> **Note**
>
> - Die SOAP API unterstützt kein S3-Versioning. Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt.
> - Für jede Version eines gespeicherten und übertragenen Objekts gelten die Standardpreise von Amazon S3. Jede Version eines Objekts besteht aus dem vollständigen Objekt, nicht nur aus einem Delta gegenüber der vorherigen Version. Wenn Sie also drei Versionen eines Objekts gespeichert haben, fallen Gebühren für die drei Objekte an.

Nicht versionierte, versionings-fähige und Buckets mit ausgesetztem Versioning

Buckets können einen von drei Zuständen aufweisen:

- Nicht versioniert (Standard)
- Versioningsfähig

- Ausgesetztes Versioning

Sie aktivieren und unterbrechen das Versioning auf Bucket-Ebene. Nachdem Sie einen Bucket versionsfähig gemacht haben, kann er nicht in einen nicht versionsfähigen Status zurückgesetzt werden. Sie können das Versioning für diesen Bucket jedoch aussetzen.

Der Versioning-Status gilt für alle (niemals für eine Untermenge) der Objekte in diesem Bucket. Wenn Sie die Versioning in einem Bucket aktivieren, werden alle neuen Objekte versioniert und mit einer eindeutigen Versions-ID versehen. Objekte, die zum Zeitpunkt der Aktivierung der Versioning im Bucket bereits im Bucket vorhanden waren, werden immer versioniert und erhalten eine eindeutige Versions-ID, wenn sie durch zukünftige Anforderungen geändert werden. Beachten Sie Folgendes:

- Objekte, die in Ihrem Bucket gespeichert waren, bevor Sie den Versioning-Status einrichten, haben die Versions-ID `null`. Wenn Sie das Versioning aktivieren, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht mehr. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anfragen verarbeitet. Weitere Informationen finden Sie unter Arbeiten mit Objekten in einem versioning-fähigen Bucket.
- Der Bucket-Eigentümer (oder ein anderer Benutzer mit geeigneten Berechtigungen) kann das Versioning aussetzen, um zu verhindern, dass sich weitere Objektversionen ansammeln. Wenn Sie das Versioning aussetzen, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anfragen verarbeitet. Weitere Informationen finden Sie unter Arbeiten mit Objekten in einem Bucket mit ausgesetztem Versioning.

Verwenden des S3-Versioning mit dem S3-Lebenszyklus

Zum Anpassen Ihrer Datenaufbewahrungsmethode und zur Speicherkostenkontrolle verwenden Sie das Objekt-Versioning mit dem S3-Lebenszyklus. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus. Informationen zum Erstellen von S3-Lebenszykluskonfigurationen mit der AWS Management Console, AWS CLI, SDKs oder der REST-API finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket. AWS SDKs

> **Important**
>
> Wenn in dem nicht-versionierten Bucket eine Lebenszyklus-Konfiguration für den Ablauf von Objekten vorhanden ist und Sie dasselbe Verhalten hinsichtlich einer dauerhaften Löschung beim Aktivieren der Versionsverwaltung beibehalten möchten, müssen Sie eine Konfiguration für den Ablauf von nicht aktuellen Objekten hinzufügen. Die Lebenszyklus-Konfiguration

für den Ablauf von nicht aktuellen Objekten verwaltet das Löschen der nicht aktuellen Objektversionen in dem Bucket mit aktivierter Versionsverwaltung. (Ein versionsfähiges Bucket behält eine kurzfristige und null oder mehr langfristige Objektversionen.) Weitere Informationen finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

Informationen zum Arbeiten mit S3-Versioning finden Sie in den folgenden Themen.

Themen

- Wie funktioniert S3-Versioning
- Aktivieren des Versioning für Buckets
- Konfigurieren von MFA Delete
- Arbeiten mit Objekten in einem versioning-fähigen Bucket
- Arbeiten mit Objekten in einem Bucket mit ausgesetztem Versioning

Wie funktioniert S3-Versioning

Sie können die S3-Versionsverwaltung verwenden, um mehrere Versionen eines Objekts in einem Bucket zu behalten und so Objekte wiederherzustellen, die versehentlich gelöscht oder überschrieben wurden. Wenn Sie beispielsweise die S3-Versionsverwaltung auf einen Bucket anwenden, werden die folgenden Änderungen vorgenommen:

- Wenn Sie ein Objekt löschen, fügt Amazon S3 eine Löschmarkierung hinzu, statt das Objekt dauerhaft zu entfernen. Die Löschmarkierung wird zur aktuellen Objektversion. Sie können dann die vorherige Version wiederherstellen. Weitere Informationen finden Sie unter Löschen von Objekten aus einem versioning-fähigen Bucket.
- Wenn Sie ein Objekt überschreiben, fügt Amazon S3 eine neue Objektversion im Bucket hinzu. Die vorherige Version verbleibt im Bucket und wird zu einer nicht aktuellen Version. Sie können die vorherige Version wiederherstellen.

> **Note**
>
> Für jede Version eines gespeicherten und übertragenen Objekts gelten die Standardpreise von Amazon S3. Jede Version eines Objekts besteht aus dem vollständigen Objekt, nicht aus

einem Delta gegenüber der vorherigen Version. Wenn Sie also drei Versionen eines Objekts gespeichert haben, fallen Gebühren für die drei Objekte an.

Jedem von Ihnen erstellten S3-Bucket ist eine Versioning-Subressource zugeordnet. (Weitere Informationen finden Sie unter Optionen für die Bucket-Konfiguration.) Standardmäßig ist Ihr Bucket nicht versioning-fähig, und dementsprechend speichert die Versioning-Subressource eine leere Versioning-Konfiguration wie folgt.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</VersioningConfiguration>
```

Um die Versionsverwaltung zu aktivieren, können Sie eine Anfrage an Amazon S3 mit einer Versionskonfiguration senden, die einen `Enabled`-Status enthält.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

Um das Versioning auszusetzen, setzen Sie den Statuswert auf `Suspended`.

> **Note**
>
> Wenn Sie die Versionsverwaltung für einen Bucket zum ersten Mal aktivieren, kann es einen Moment dauern, bis die Änderung vollständig verbreitet ist. Wir empfehlen, dass Sie nach dem Aktivieren der Versionsverwaltung 15 Minuten warten, bevor Sie Schreibvorgänge (`PUT` oder `DELETE`) für Objekte im Bucket ausführen.

Der Bucket-Eigentümer und alle autorisierten AWS Identity and Access Management (IAM)-Benutzer können das Versioning aktivieren. Der Bucket-Eigentümer ist der AWS-Konto , der den Bucket erstellt hat. Weitere Informationen zu Berechtigungen finden Sie unter Identity and Access Management in Amazon S3.

Weitere Informationen zum Aktivieren und Deaktivieren der S3-Versionsverwaltung mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) oder REST-API finden Sie unter the section called “Aktivieren des Versioning für Buckets”.

Themen

- Versions-ID
- Versioning-Workflows

Versions-ID

Wenn Sie das Versioning für einen Bucket aktivieren, generiert Amazon S3 automatisch eine eindeutige Versions-ID für das Objekt, das gespeichert wird. Beispielsweise könnten Sie in einem Bucket zwei Objekte mit demselben Schlüssel (Objektname) haben, aber mit unterschiedlichen Versions-IDs, wie beispielsweise `photo.gif` (Version 111111) und `photo.gif` (Version 121212).

Jedes Objekt hat eine Versions-ID, unabhängig davon, ob die S3-Versionierung aktiviert ist oder nicht. Wenn die S3-Versionsverwaltung nicht aktiviert ist, legt Amazon S3 den Wert der Versions-ID auf `null` fest. Wenn das S3-Versioning aktiviert ist, weist Amazon S3 dem Objekt einen Versions-ID-Wert zu. Dieser Wert unterscheidet dieses Objekt von anderen Versionen desselben Schlüssels.

Wenn Sie das S3-Versioning für einen vorhandenen Bucket aktivieren, bleiben Objekte, die bereits im Bucket gespeichert sind, unverändert. Die Version-IDs (`null`), Inhalte und Berechtigungen bleiben unverändert. Nachdem Sie die S3-Versionsverwaltung aktiviert haben, erhält jedes Objekt, das dem Bucket hinzugefügt wird, eine Versions-ID, die sie von anderen Versionen desselben Schlüssels unterscheidet.

Nur Amazon S3 generiert Versions-IDs und diese können nicht bearbeitet werden. Versions-IDs sind Unicode-, UTF-8-codierte, URL-fähige, nicht einsichtige Zeichenfolgen, die nicht mehr als 1.024 Byte lang sind. Im Folgenden wird ein Beispiel gezeigt:

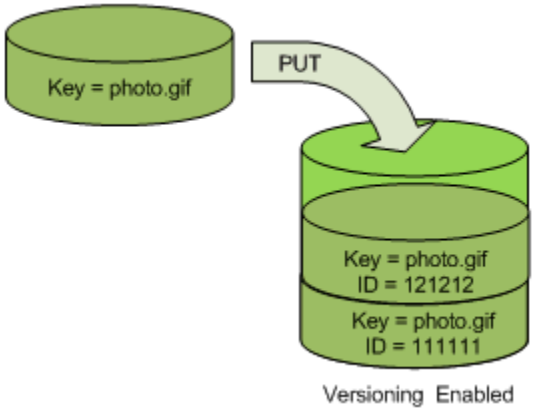
```
3sL4kqtJlcpXroDTDmJ+rmSpXd3dIbrHY+MTRCxf3vjVBH40Nr8X8gdRQBpUMLUo
```

> **Note**
>
> Zur Vereinfachung verwenden die anderen Beispiele in diesem Thema viel kürzere IDs.

Versioning-Workflows

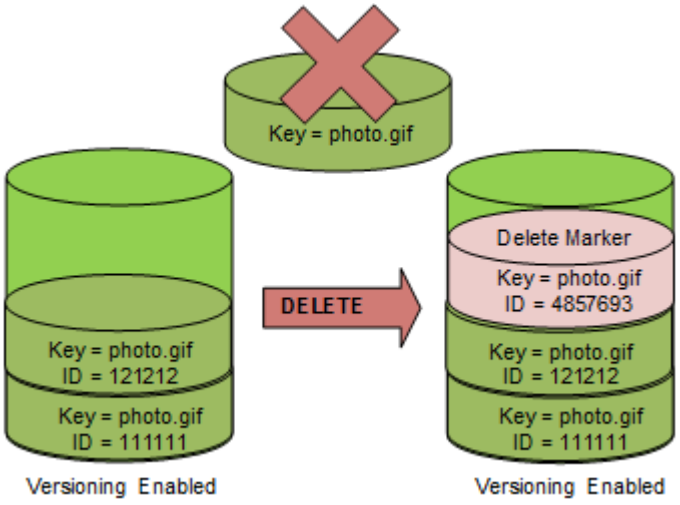
Wenn Sie mit PUT ein Objekt in einen Bucket mit aktiviertem Versioning schreiben, wird die nicht aktuelle Version nicht überschrieben. Wie in der folgenden Abbildung dargestellt, tritt das folgende Verhalten auf, wenn eine neue Version von `photo.gif` mit PUT in einen Bucket geschrieben wird, der bereits ein Objekt desselben Namens enthält:

- Das ursprüngliche Objekt (ID = 111111) verbleibt im Bucket.
- Amazon S3 generiert eine neue Versions-ID (121212) und fügt diese neuere Version des Objekts dem Bucket hinzu.

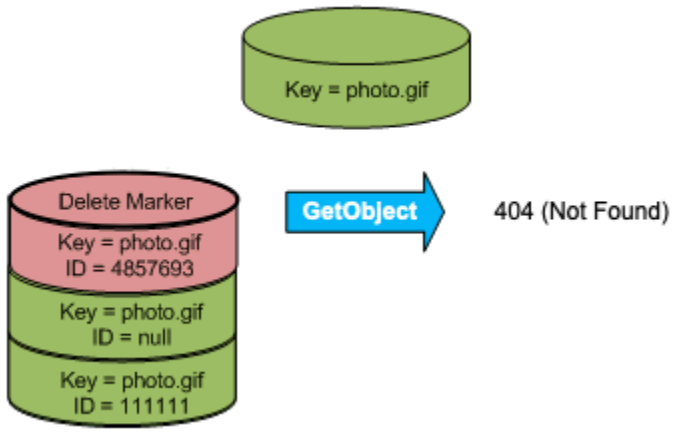


Mit dieser Funktion können Sie eine vorherige Version eines Objekts abrufen, wenn ein Objekt versehentlich überschrieben oder gelöscht wurde.

Wenn Sie DELETE für ein Objekt ausführen, bleiben alle Versionen in dem Bucket, und Amazon S3 fügt eine Löschmarkierung ein, wie in der folgenden Abbildung gezeigt.

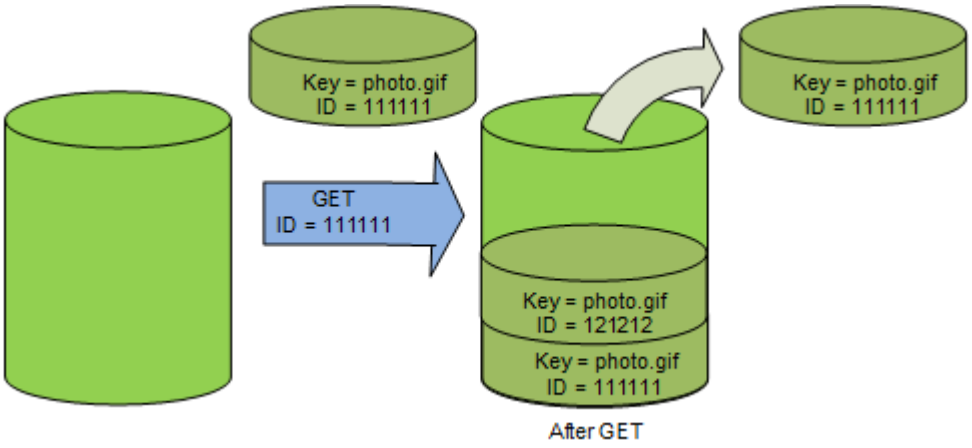


Die Löschmarkierung wird zur aktuellen Version des Objekts. Standardmäßig wird durch GET-Anforderungen die zuletzt gespeicherte Version abgerufen. Eine `GET Object`-Anforderung gibt einen `404 Not Found`-Fehler zurück, wenn die aktuelle Version eine Löschmarkierung ist, wie in der folgenden Abbildung gezeigt.

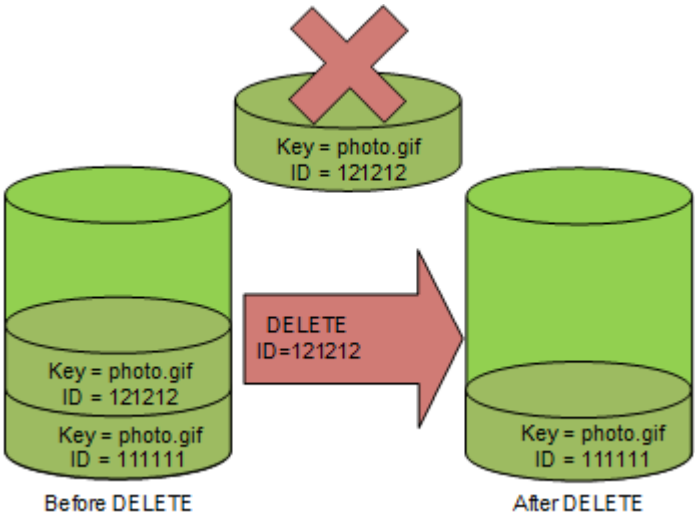


Sie können jedoch mit GET eine nicht aktuelle Version eines Objekts abrufen, indem Sie ihre Versions-ID angeben. In der folgenden Abbildung wird mit GET eine spezifische Objektversion abgerufen, 111111. Amazon S3 gibt diese Objektversion zurück, auch wenn es nicht die aktuelle Version ist.

Weitere Informationen finden Sie unter Abrufen von Objektversionen aus einem versioning-fähigen Bucket.



Sie können ein Objekt permanent löschen, indem Sie die Version angeben, die Sie löschen wollen. Nur der Eigentümer eines Amazon-S3-Buckets kann eine Version permanent löschen. Wenn Ihre DELETE-Operation die `versionId` angibt, wird diese Objektversion dauerhaft gelöscht, und Amazon S3 fügt keine Löschmarkierung ein.



Sie können die Sicherheit erhöhen, indem Sie einen Bucket mit aktivierter Multi-Faktor-Authentifizierung (MFA) Delete konfigurieren. In diesem Fall muss der Bucket-Eigentümer zwei Authentifizierungsformen in jede Anforderung aufnehmen, um eine Version zu löschen oder den Versionsverwaltungsstatus des Buckets zu ändern. Weitere Informationen finden Sie unter Konfigurieren von MFA Delete.

Wann werden neue Versionen für ein Objekt erstellt?

Neue Versionen werden nur erstellt, wenn Sie für ein neues Objekt PUT ausführen. Beachten Sie, dass bestimmte Aktionen wie `CopyObject` durch die Implementierung einer PUT-Operation funktionieren.

Einige Aktionen, die das aktuelle Objekt ändern, erstellen keine neue Version, da sie kein neues Objekt mit PUT bearbeiten. Dies umfasst Aktionen wie das Ändern der Markierungen für ein Objekt.

> **Important**
>
> Wenn Sie eine deutliche Zunahme von HTTP 503-Antworten (Service nicht verfügbar) feststellen, die für Amazon-S3-PUT- oder `DELETE`-Objektanfragen an einen Bucket mit aktivierter S3-Versionsverwaltung eingehen, befinden sich möglicherweise ein oder mehrere Objekte im Bucket, für die Millionen von Versionen vorhanden sind. Weitere Informationen finden Sie im Abschnitt zur S3-Versionsverwaltung unter Fehlerbehebung.

Aktivieren des Versioning für Buckets

Sie können Amazon-S3-Versioning nutzen, um mehrere Versionen eines Objekts in einem Bucket aufzubewahren. Dieser Abschnitt enthält Beispiele für die Aktivierung des Versionings für einen Bucket mithilfe der Konsole, der REST API, der AWS SDKs und AWS Command Line Interface (AWS CLI).

> **Note**
>
> Wenn Sie das Versioning für einen Bucket zum ersten Mal aktivieren, kann es bis zu 15 Minuten dauern, bis die Änderung vollständig verbreitet ist. Wir empfehlen, dass Sie nach dem Aktivieren des Versioning 15 Minuten warten, bevor Sie Schreibvorgänge (PUT oder DELETE) für Objekte im Bucket ausführen. Schreibvorgänge, die vor Abschluss dieser Konvertierung ausgegeben wurden, können für nicht versionierte Objekte gelten.

Weitere Informationen über das S3-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets. Informationen zum Arbeiten mit Objekten, die sich in einem versioning-fähigen Bucket befinden, finden Sie unter Arbeiten mit Objekten in einem versioning-fähigen Bucket.

Weitere Informationen zur Verwendung von S3 Versioning zum Schutz von Daten finden Sie im Tutorial: Schutz von Daten in Amazon S3 vor versehentlichem Löschen oder Anwendungsfehlern mithilfe von S3 Versioning, S3 Object Lock und S3 Replication.

Jedem von Ihnen erstellten S3-Bucket ist eine Versioning-Subressource zugeordnet. (Weitere Informationen finden Sie unter Optionen für die Bucket-Konfiguration.) Standardmäßig ist Ihr Bucket nicht versioning-fähig, und dementsprechend speichert die Versioning-Subressource eine leere Versioning-Konfiguration wie folgt.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</VersioningConfiguration>
```

Um das Versioning zu aktivieren, können Sie eine Anfrage an Amazon S3 mit einer Versionskonfiguration senden, die einen Status enthält.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

Um das Versioning auszusetzen, setzen Sie den Statuswert auf `Suspended`.

Der Bucket-Eigentümer und alle autorisierten Benutzer können das Versioning aktivieren. Der Bucket-Eigentümer ist der AWS-Konto , der den Bucket erstellt hat (das Stammkonto). Weitere Informationen zu Berechtigungen finden Sie unter Identity and Access Management in Amazon S3.

In den folgenden Abschnitten finden Sie weitere Informationen zum Aktivieren der S3-Versionsverwaltung mithilfe der Konsole AWS CLI, und der AWS SDKs .

Verwenden der S3-Konsole

Gehen Sie wie folgt vor, um die AWS Management Console zum Aktivieren des Versionings für einen S3-Bucket zu verwenden.

Aktivieren und Deaktivieren des Versionings für einen S3-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie Versioning aktivieren möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Bucket Versioning (Bucket-Versioning) die Option Edit (Bearbeiten).
5. Wählen Sie Suspend (Anhalten) oder Enable (Aktivieren) und dann Save changes (Änderungen speichern).

> **Note**
>
> Sie können die AWS Multi-Faktor-Authentifizierung (MFA) mit Versioning verwenden. Wenn Sie MFA mit Versioning verwenden, müssen Sie AWS-Konto die Zugriffsschlüssel Ihres und einen gültigen Code vom MFA-Gerät des Kontos angeben, um eine Objektversion dauerhaft zu löschen oder das Versioning zu unterbrechen oder zu reaktivieren.
> Um MFA mit Versioning zu verwenden, aktivieren Sie `MFA Delete`. Sie können jedoch `MFA Delete` nicht mit der AWS Management Console aktivieren. Sie müssen die AWS Command Line Interface (AWS CLI) oder die -API verwenden. Weitere Informationen finden Sie unter Konfigurieren von MFA Delete.

Verwenden der AWS CLI

Im folgenden Beispiel wird das Versioning auf einem S3-Bucket aktiviert.

```
aws s3api put-bucket-versioning --bucket DOC-EXAMPLE-BUCKET1 --versioning-configuration
 Status=Enabled
```

Das folgende Beispiel ermöglicht das Löschen der S3-Versionsverwaltung und Multi-Faktor Authentifizierung (MFA) für einen Bucket.

```
aws s3api put-bucket-versioning --bucket DOC-EXAMPLE-BUCKET1 --versioning-configuration
 Status=Enabled,MFADelete=Enabled --mfa "SERIAL 123456"
```

> **Note**
>
> Die Verwendung von MFA Delete erfordert ein genehmigtes physisches oder virtuelles Authentifizierungsgerät. Weitere Informationen zur Verwendung von MFA Delete in Amazon S3 finden Sie unter Konfigurieren von MFA Delete.

Weitere Informationen zum Aktivieren des Versionings mithilfe der finden Sie AWS CLI unter put-bucket-versioning in der AWS CLI -Befehlsreferenz.

Verwenden der AWS SDKs

Die folgenden Beispiele aktivieren das Versioning für einen Bucket und rufen dann den Versioning-Status mithilfe der AWS SDK for Java und der ab AWS SDK for .NET. Informationen zur Verwendung anderer AWS -SDKs finden Sie im AWS -Entwicklerzentrum.

.NET

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using System;
using Amazon.S3;
using Amazon.S3.Model;
```

```
namespace s3.amazon.com.docsamples
{
    class BucketVersioningConfiguration
    {
        static string bucketName = "*** bucket name ***";

        public static void Main(string[] args)
        {
            using (var client = new AmazonS3Client(Amazon.RegionEndpoint.USEast1))
            {
                try
                {
                    EnableVersioningOnBucket(client);
                    string bucketVersioningStatus =
 RetrieveBucketVersioningConfiguration(client);
                }
                catch (AmazonS3Exception amazonS3Exception)
                {
                    if (amazonS3Exception.ErrorCode != null &&
                        (amazonS3Exception.ErrorCode.Equals("InvalidAccessKeyId")
                        ||
                        amazonS3Exception.ErrorCode.Equals("InvalidSecurity")))
                    {
                        Console.WriteLine("Check the provided AWS Credentials.");
                        Console.WriteLine(
                        "To sign up for service, go to http://aws.amazon.com/s3");
                    }
                    else
                    {
                        Console.WriteLine(
                         "Error occurred. Message:'{0}' when listing objects",
                         amazonS3Exception.Message);
                    }
                }
            }

            Console.WriteLine("Press any key to continue...");
            Console.ReadKey();
        }

        static void EnableVersioningOnBucket(IAmazonS3 client)
        {
```

```
                PutBucketVersioningRequest request = new PutBucketVersioningRequest
                {
                    BucketName = bucketName,
                    VersioningConfig = new S3BucketVersioningConfig
                    {
                        Status = VersionStatus.Enabled
                    }
                };

                PutBucketVersioningResponse response =
 client.PutBucketVersioning(request);
        }


        static string RetrieveBucketVersioningConfiguration(IAmazonS3 client)
        {
                GetBucketVersioningRequest request = new GetBucketVersioningRequest
                {
                    BucketName = bucketName
                };

                GetBucketVersioningResponse response =
 client.GetBucketVersioning(request);
                return response.VersioningConfig.Status;
            }
    }
}
```

Java

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Region;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.SetBucketVersioningConfigurationRequest;
```

```
public class BucketVersioningConfigurationExample {
    public static String bucketName = "*** bucket name ***";
    public static AmazonS3Client s3Client;

    public static void main(String[] args) throws IOException {
        s3Client = new AmazonS3Client(new ProfileCredentialsProvider());
        s3Client.setRegion(Region.getRegion(Regions.US_EAST_1));
        try {

            // 1. Enable versioning on the bucket.
         BucketVersioningConfiguration configuration =
           new BucketVersioningConfiguration().withStatus("Enabled");

   SetBucketVersioningConfigurationRequest setBucketVersioningConfigurationRequest
 =
     new SetBucketVersioningConfigurationRequest(bucketName,configuration);


 s3Client.setBucketVersioningConfiguration(setBucketVersioningConfigurationRequest);

   // 2. Get bucket versioning configuration information.
   BucketVersioningConfiguration conf =
 s3Client.getBucketVersioningConfiguration(bucketName);
    System.out.println("bucket versioning configuration status:    " +
 conf.getStatus());

        } catch (AmazonS3Exception amazonS3Exception) {
            System.out.format("An Amazon S3 error occurred. Exception: %s",
 amazonS3Exception.toString());
        } catch (Exception ex) {
            System.out.format("Exception: %s", ex.toString());
        }
    }
}
```

Python

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwendung der AWS SDK for Python (Boto).

Das folgende Python-Beispiel erstellt einen Amazon-S3-Bucket, aktiviert ihn für das Versioning und konfiguriert einen Lebenszyklus, der nicht-aktuelle Objektversionen nach 7 Tagen ablaufen lässt.

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
 lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                   configured lifecycle rules.
    :return: The newly created bucket.
    """
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
```

```
        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
expiration},
                    }
                ]
            }
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
        )
    except ClientError as error:
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket
```

Konfigurieren von MFA Delete

Wenn Sie in Amazon-S3-Buckets mit S3-Versioning arbeiten, können Sie optional eine weitere Sicherheitsebene hinzufügen, indem Sie einen Bucket konfigurieren, um MFA (multi-factor authentication) delete zu aktivieren. In diesem Fall muss der Bucket-Eigentümer zwei Authentifizierungsformen in jede Anforderung aufnehmen, um eine Version zu löschen oder den Versioning-Status des Buckets zu ändern.

Die MFA-Löschfunktion erfordert eine zusätzliche Authentifizierung für die folgenden Vorgänge:

- Ändern des Versioning-Status Ihres Buckets

- Dauerhaftes Löschen einer Objektversion

MFA Delete fordert zwei Authentifizierungsformen in Kombination:

- Ihre Sicherheitsanmeldeinformationen
- Die Verkettung einer gültigen Seriennummer, eines Leerzeichens und des sechsstelligen Codes, der auf einem zugelassenen Authentifizierungsgerät angezeigt wird

MFA Delete bietet damit zusätzliche Sicherheit, wenn beispielsweise Ihre Sicherheitsanmeldeinformationen nicht mehr vertrauenswürdig sind. MFA Delete kann dazu beitragen, versehentliche Bucket-Löschungen zu verhindern, indem der Benutzer, der die Löschaktion einleitet, verpflichtet wird, den physischen Besitz eines MFA-Geräts mit einem MFA-Code nachzuweisen und der Löschaktion eine zusätzliche Sicherheitsschicht hinzuzufügen.

Um Buckets zu identifizieren, für die MFA Delete aktiviert ist, können Sie Metriken von Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Weitere Informationen finden Sie unter Bewertung Ihrer Speicheraktivität und -nutzung mit S3 Storage Lens. Eine vollständige Liste der Metriken finden Sie im Glossar der S3-Storage-Lens-Metriken.

Der Bucket-Eigentümer, die AWS-Konto , die den Bucket erstellt hat (Root-Konto), und alle autorisierten Benutzer können das Versioning aktivieren. Allerdings kann nur der Bucket-Eigentümer (Root-Konto) MFA Delete aktivieren. Weitere Informationen finden Sie unter Sichern des Zugriffs auf AWS mithilfe von MFA im - AWS Sicherheitsblog.

> **Note**
>
> Um MFA Delete mit Versioning zu verwenden, aktivieren Sie `MFA Delete`. Sie können jedoch nicht `MFA Delete` mit der AWS Management Console aktivieren. Sie müssen die AWS Command Line Interface (AWS CLI) oder die -API verwenden.
> Beispiele für die Verwendung von MFA Delete mit Versioning finden Sie im Abschnitt "Beispiele" im Thema Aktivieren des Versioning für Buckets.
> Sie können MFA Löschen nicht mit Lebenszykluskonfigurationen verwenden. Weitere Informationen zu Lebenszykluskonfigurationen und deren Interaktion mit anderen Konfigurationen finden Sie unter Lebenszyklus- und andere Bucket-Konfigurationen.

Für das Aktivieren oder Deaktivieren von MFA Delete verwenden Sie dieselbe API, die Sie für die Konfiguration des Versionings für einen Bucket verwenden. Amazon S3 speichert die MFA-Delete-Konfiguration in derselben Versioning-Subressource, in der auch der Versioning-Status des Buckets gespeichert ist.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>VersioningState</Status>
  <MfaDelete>MfaDeleteState</MfaDelete>
</VersioningConfiguration>
```

Um MFA Delete zu nutzen, verwenden Sie ein Hardwaregerät oder ein virtuelles MFA-Gerät, um einen Authentifizierungscode zu generieren. Das folgende Beispiel zeigt einen generierten Authentifizierungscode, angezeigt auf einem Hardwaregerät.

MFA Delete und ein durch MFA geschützter API-Zugriff sind Funktionen, die Schutz in bestimmten Situationen bieten sollen. Sie konfigurieren MFA Delete für einen Bucket, um sicherzustellen, dass die Daten in Ihrem Bucket nicht versehentlich gelöscht werden können. Der durch MFA geschützte API-Zugriff wird verwendet, um einen zusätzlichen Authentifizierungsfaktor (MFA-Code) einzuführen, wenn Sie auf sensible Amazon-S3-Ressourcen zugreifen. Sie können für alle Vorgänge für Amazon-S3-Ressourcen fordern, dass sie mit temporären Anmeldeinformationen ausgeführt werden, die mit MFA erstellt wurden. Ein Beispiel finden Sie unter Verlangen von MFA.

Weitere Informationen zum Kauf und zur Aktivierung eines Authentifizierungsgeräts finden Sie unter Multi-Faktor-Authentifizierung.

So aktivieren Sie das Löschen der S3-Versionsverwaltung und von MFA

Verwenden der AWS CLI

Das folgende Beispiel ermöglicht das Löschen der S3-Versionsverwaltung und Multi-Faktor Authentifizierung (MFA) für einen Bucket.

```
aws s3api put-bucket-versioning --bucket DOC-EXAMPLE-BUCKET1 --versioning-configuration
 Status=Enabled,MFADelete=Enabled --mfa "SERIAL 123456"
```

Verwenden der REST-API

Weitere Informationen zum Angeben von MFA Delete mit der Amazon S3-REST-API finden Sie unter PutBucketVersioning Amazon Simple Storage Service API Reference .

Arbeiten mit Objekten in einem versioning-fähigen Bucket

Objekte, die in einem Amazon-S3-Bucket gespeichert waren, bevor Sie den Versioning-Status einrichten, haben die Versions-ID `null`. Wenn Sie das Versioning aktivieren, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht mehr. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anfragen verarbeitet.

Übergang von Objektversionen

Sie können Lebenszyklus-Konfigurationsregeln für Objekte definieren, die einen definierten Lebenszyklus haben, um Objektversionen zu einem bestimmten Zeitpunkt in der Lebensdauer des Objekts in die Speicherklasse S3 Glacier Flexible Retrieval zu überführen. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Die Themen in diesem Abschnitt erklären verschiedene Objekt-Vorgänge in einem Bucket mit aktiviertem Versioning. Weitere Informationen über das Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

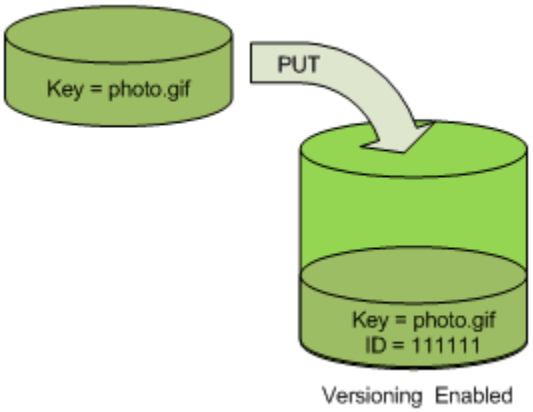
Themen

- Hinzufügen von Objekten zu versioning-fähigen Buckets
- Auflisten von Objekten in einem versioning-fähigen Bucket
- Abrufen von Objektversionen aus einem versioning-fähigen Bucket
- Löschen von Objekten aus einem versioning-fähigen Bucket
- Konfigurieren von versionierten Objektberechtigungen

Hinzufügen von Objekten zu versioning-fähigen Buckets

Nachdem Sie das Versioning für einen Bucket aktiviert haben, fügt Amazon S3 jedem im Bucket gespeicherten Objekt automatisch eine eindeutige Versions-ID hinzu (mit `PUT`, `POST` oder `CopyObject`).

Die folgende Abbildung zeigt, dass Amazon S3 jedem Objekt automatisch eine eindeutige Versions-ID hinzufügt, wenn es einem versioningfähigen Bucket hinzugefügt wird.



> **Note**
>
> Die von Amazon S3 zugewiesenen Versions-ID-Werte sind URL-sicher (können als Teil einer URI angegeben werden).

Weitere Informationen über das Versioning finden Sie unter Verwenden der Versioning in S3-Buckets. Sie können Objektversionen zu einem versionsfähigen Bucket mithilfe der Konsole, AWS SDKs und REST API hinzufügen.

Verwenden der Konsole

Detaillierte Anweisungen finden Sie unter Objekte hochladen.

Verwenden der AWS -SDKs

Beispiele für das Hochladen von Objekten mit den - AWS SDKs für Java, .NET und PHP finden Sie unter Objekte hochladen. Die Beispiele für das Hochladen von Objekten in versioning-fähige und nicht versioning-fähige Buckets sind gleich, aber bei versioning-fähigen Buckets weist Amazon S3 eine Versionsnummer zu. Andernfalls ist die Versionsnummer null.

Informationen zur Verwendung anderer AWS SDKs finden Sie im AWS -Entwicklerzentrum.

Verwenden der REST-API

Hinzufügen von Objekten zu versioning-fähigen Buckets

1. Aktivieren des Versionings für einen Bucket mit einer `PutBucketVersioning`-Anforderung.

Weitere Informationen finden Sie unter PutBucketVersioning in der API-Referenz zu Amazon Simple Storage Service.

2. Senden Sie eine `PUT`-, `POST`- oder `CopyObject`-Anforderung, um ein Objekt im Bucket zu speichern.

Wenn Sie einem versioning-fähigen Bucket ein Objekt hinzufügen, gibt Amazon S3 die Versions-ID des Objekts im Antwort-Header `x-amz-version-id` zurück, wie im folgenden Beispiel gezeigt.

```
x-amz-version-id: 3/L4kqtJlcpXroDTDmJ+rmSpXd3dIbrHY
```

Auflisten von Objekten in einem versioning-fähigen Bucket

Dieser Abschnitt zeigt Beispiele, wie Objektversionen aus einem versioning-fähigen Bucket aufgelistet werden. Amazon S3 speichert Versionsinformationen zu Objekten in der versions-Subressource, die dem Bucket zugeordnet ist. Weitere Informationen finden Sie unter Optionen für die Bucket-Konfiguration. Um die Objekte in einem versionsfähigen Bucket aufzulisten, benötigen Sie die `ListBucketVersions`-Berechtigung.

Verwenden der S3-Konsole

Befolgen Sie diese Schritte, um die Amazon-S3-Konsole zu verwenden, um die verschiedenen Versionen eines Objekts anzuzeigen.

Mehrere Versionen eines Objekts anzeigen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
3. Um eine Liste der Versionen der Objekte in dem Bucket anzuzeigen, wählen Sie den Schalter Show versions (Versionen anzeigen).

   Die Konsole zeigt für jede Objektversion eine eindeutige Versions-ID, das Datum und die Uhrzeit, wann das Objekt erstellt wurde, sowie weitere Eigenschaften an. (Objekte, die in Ihrem Bucket gespeichert waren, bevor Sie den Versioning-Status einrichten, haben die Versions-ID null.)

   Um die Objekte ohne die Versionen aufzulisten, wählen Sie den Schalter List versions (Versionen auflisten) .

Objektversionen können auch in der Objektübersicht auf der Konsole angezeigt, heruntergeladen und gelöscht werden. Weitere Informationen finden Sie unter Anzeigen einer Objektübersicht in der Amazon-S3-Konsole.

> **Note**
>
> Um auf Objektversionen zuzugreifen, die älter als 300 Versionen sind, müssen Sie die AWS CLI oder die URL des Objekts verwenden.

> **Important**
>
> Sie können den Löschvorgang für ein Objekt nur rückgängig machen, wenn seine aktuelle Version gelöscht wurde. Es ist nicht möglich, das Löschen einer vorherigen Version eines Objekts rückgängig zu machen, das gelöscht wurde. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Verwenden der AWS SDKs

Die Beispiele in diesem Abschnitt veranschaulichen, wie Sie eine Objektliste aus einem versioning-fähigen Bucket abrufen. Jede Anforderung gibt bis zu 1000 Versionen zurück, sofern Sie keine kleinere Anzahl festlegen. Wenn die Versionen im Bucket dieses Limit überschreiten, senden Sie eine Reihe von Anforderungen, um die Liste aller Versionen abzurufen. Dieser Vorgang zur „seitenweisen“ Rückgabe von Ergebnissen wird als Paginierung bezeichnet.

Um zu veranschaulichen, wie Paginierung funktioniert, limitieren die Beispiele jede Antwort auf zwei Objektversionen. Nachdem die erste Seite mit Ergebnissen abgerufen wurde, wird in jedem Beispiel überprüft, ob die Versionsliste abgeschnitten wurde. Wurde sie abgeschnitten, dann wird im Beispiel mit dem seitenweisen Abruf fortgefahren, bis alle Versionen abgerufen wurden.

> **Note**
>
> Die folgenden Beispiele funktionieren auch bei einem Bucket, der versioning-fähig ist, oder bei Objekten ohne individuelle Versionen. In solchen Fällen gibt Amazon S3 die Objektliste mit der Versions-ID von `null` zurück.

Informationen zur Verwendung anderer AWS SDKs finden Sie im AWS -Entwicklerzentrum.

Java

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListVersionsRequest;
import com.amazonaws.services.s3.model.S3VersionSummary;
import com.amazonaws.services.s3.model.VersionListing;

public class ListKeysVersioningEnabledBucket {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Retrieve the list of versions. If the bucket contains more versions
            // than the specified maximum number of results, Amazon S3 returns
            // one page of results per request.
            ListVersionsRequest request = new ListVersionsRequest()
                    .withBucketName(bucketName)
                    .withMaxResults(2);
            VersionListing versionListing = s3Client.listVersions(request);
            int numVersions = 0, numPages = 0;
            while (true) {
                numPages++;
                for (S3VersionSummary objectSummary :
 versionListing.getVersionSummaries()) {
                    System.out.printf("Retrieved object %s, version %s\n",
                            objectSummary.getKey(),
                            objectSummary.getVersionId());
                    numVersions++;
```

```
                }
                // Check whether there are more pages of versions to retrieve. If
                // there are, retrieve them. Otherwise, exit the loop.
                if (versionListing.isTruncated()) {
                    versionListing =
 s3Client.listNextBatchOfVersions(versionListing);
                } else {
                    break;
                }
            }
            System.out.println(numVersions + " object versions retrieved in " +
 numPages + " pages");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ListObjectsVersioningEnabledBucketTest
    {
        static string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
```

```
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main(string[] args)
        {
            s3Client = new AmazonS3Client(bucketRegion);
            GetObjectListWithAllVersionsAsync().Wait();
        }

        static async Task GetObjectListWithAllVersionsAsync()
        {
            try
            {
                ListVersionsRequest request = new ListVersionsRequest()
                {
                    BucketName = bucketName,
                    // You can optionally specify key name prefix in the request
                    // if you want list of object versions of a specific object.

                    // For this example we limit response to return list of 2
versions.
                    MaxKeys = 2
                };
                do
                {
                    ListVersionsResponse response = await
s3Client.ListVersionsAsync(request);
                    // Process response.
                    foreach (S3ObjectVersion entry in response.Versions)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }

                    // If response is truncated, set the marker to get the next
                    // set of keys.
                    if (response.IsTruncated)
                    {
                        request.KeyMarker = response.NextKeyMarker;
                        request.VersionIdMarker = response.NextVersionIdMarker;
                    }
                    else
                    {
```

```
                        request = null;
                    }
                } while (request != null);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Verwenden der REST-API

Example – Alle Objektversionen in einem Bucket auflisten

Um alle Versionen aller Objekte in einem Bucket aufzulisten, verwenden Sie die `versions`-Subressource in einer `GET Bucket`-Anfrage. Amazon S3 kann maximal 1000 Objekte abrufen. Jede Objektversion zählt als vollständiges Objekt. Wenn ein Bucket also zwei Schlüssel enthält (z. B. `photo.gif` und `picture.jpg`) und der erste Schlüssel 990 Versionen und der zweite Schlüssel 400 Versionen hat, ruft eine einzelne Abfrage alle 990 Versionen von `photo.gif` und nur die 10 neuesten Versionen von `picture.jpg` ab.

Amazon S3 gibt Objektversionen in der Reihenfolge zurück, in der sie gespeichert wurden, wobei die zuletzt gespeicherte zuerst zurückgegeben wird.

Geben Sie in einer `GET Bucket`-Anforderung die `versions`-Subressource an.

```
GET /?versions HTTP/1.1
Host: bucketName.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 +0000
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Example – Abruf aller Versionen eines Schlüssels

Wenn Sie eine Untermenge von Objektversionen abrufen möchten, verwenden Sie die Anforderungsparameter für `GET Bucket`. Weitere Informationen finden Sie unter GET Bucket.

1. Setzen Sie den Parameter `prefix` auf den Schlüssel des Objekts, das Sie abrufen wollen.
2. Senden Sie eine `GET Bucket`-Anforderung unter Verwendung der `versions`-Subressource und `prefix`.

   ```
   GET /?versions&prefix=objectName HTTP/1.1
   ```

Example – Abrufen von Objekten unter Verwendung eines Präfix

Das folgende Beispiel ruft Objekte ab, deren Schlüssel ist oder damit beginn `myObject`.

```
GET /?versions&prefix=myObject HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Sie können die anderen Anforderungsparameter verwenden, um eine Untermenge aller Versionen des Objekts abzurufen. Weitere Informationen finden Sie unter GET Bucket in der API-Referenz zu Amazon Simple Storage Service.

Example – Abrufen einer Liste zusätzlicher Objekte, falls die Antwort gekürzt wurde

Wenn die Anzahl der Objekte, die in einer `GET`-Anforderung zurückgegeben werden können, den Wert `max-keys` überschreitet, enthält die Antwort `<isTruncated>true</isTruncated>`, ebenso wie den ersten Schlüssel (in `NextKeyMarker`) und die erste Versions-ID (in `NextVersionIdMarker`), die die Anforderungskriterien erfüllen, aber nicht zurückgegeben wurden. Diese zurückgegebenen Werte verwenden Sie als Ausgangspunkt in einer nachfolgenden Anforderung, um die zusätzlichen Objekte abzurufen, die die `GET`-Anforderung erfüllen.

Gehen Sie wie folgt vor, um zusätzlichen Objekte abzurufen, die die ursprüngliche `GET Bucket versions`-Anforderung von einem Bucket erfüllen. Weitere Informationen zu `key-marker`, `version-id-marker`, `NextKeyMarker` und `NextVersionIdMarker` finden Sie unter GET Bucket in Amazon Simple Storage Service – API-Referenz.

Im Folgenden finden Sie zusätzliche Antworten, die die ursprüngliche `GET`-Anforderung erfüllen:

- Setzen Sie den Wert von `key-marker` auf den Schlüssel, der in `NextKeyMarker` in der vorherigen Antwort zurückgegeben wurde.
- Setzen Sie den Wert von `version-id-marker` auf die Versions-ID, die in `NextVersionIdMarker` in der vorherigen Antwort zurückgegeben wurde.
- Senden Sie eine `GET Bucket versions`-Anforderung mit `key-marker` und `version-id-marker`.

Example – Abrufen von Objekten ab einem bestimmten Schlüssel und einer bestimmten Versions-ID

```
GET /?versions&key-marker=myObject&version-id-marker=298459348571 HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Verwenden der AWS CLI

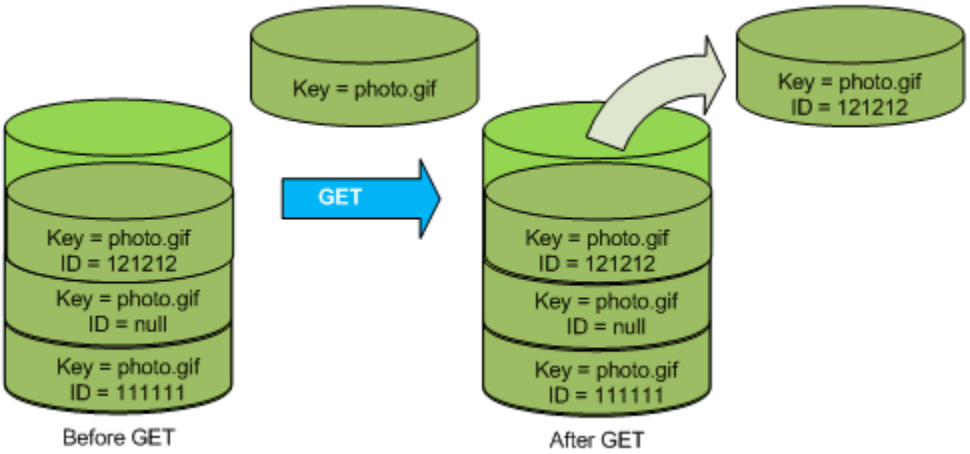
Der folgende Befehl gibt Metadaten zu allen Versionen der Objekte in einem Bucket zurück.

```
aws s3api list-object-versions --bucket DOC-EXAMPLE-BUCKET1
```

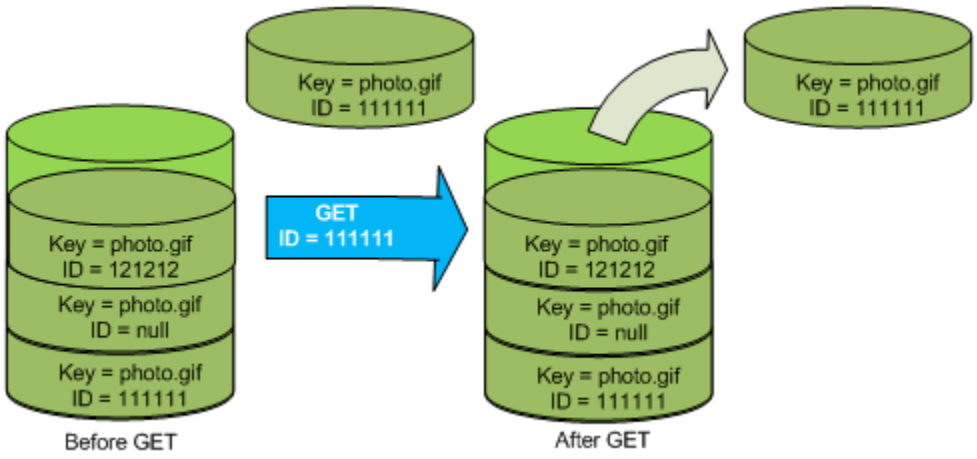
Weitere Informationen zu `list-object-versions` finden Sie unter list-object-versions in der AWS CLI -Befehlsreferenz.

Abrufen von Objektversionen aus einem versioning-fähigen Bucket

Das Versioning in Amazon S3 ist eine Möglichkeit, mehrere Varianten eines Objekts im selben Bucket zu behalten. Eine einfache `GET`-Anforderung ruft die aktuelle Version eines Objekts ab. Die folgende Abbildung zeigt, wie `GET` die aktuelle Version des Objekts zurückgibt, `photo.gif`.



Um eine spezifische Version abzurufen, müssen Sie ihre Versions-ID angeben. Die folgende Abbildung zeigt, dass eine `GET versionId`-Anforderung die angegebene Version des Objekts zurückgibt (nicht unbedingt die aktuelle).



Sie können Objektversionen in Amazon S3 über die Konsole, AWS SDKs oder die REST-API abrufen.

> **Note**
>
> Um auf Objektversionen zuzugreifen, die älter als 300 Versionen sind, müssen Sie die AWS CLI oder die URL des Objekts verwenden.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
3. Wählen Sie in der Liste Objects (Objekte) den Namen des Objekts aus.
4. Wählen Sie Versions (Versionen).

   Amazon S3 zeigt alle Versionen für das Objekt an.
5. Aktivieren Sie das Kontrollkästchen neben der Version ID (Versions-ID) für die Versionen, die Sie abrufen möchten.
6. Wählen Sie Actions (Aktionen), wählen Sie Download (Herunterladen) und speichern Sie das Objekt.

Objektversionen können auch in der Objektübersicht angezeigt, heruntergeladen und gelöscht werden. Weitere Informationen finden Sie unter Anzeigen einer Objektübersicht in der Amazon-S3-Konsole.

> **Important**
>
> Sie können den Löschvorgang für ein Objekt nur rückgängig machen, wenn seine aktuelle Version gelöscht wurde. Es ist nicht möglich, das Löschen einer vorherigen Version eines Objekts rückgängig zu machen, das gelöscht wurde. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Verwenden der AWS SDKs

Die Beispiele für das Hochladen von Objekten in nicht versioning-fähigen und versioning-fähigen Buckets sind dieselben. Für versioning-fähige Buckets weist Amazon S3 jedoch eine Versionsnummer zu. Andernfalls ist die Versionsnummer null.

Beispiele für das Herunterladen von Objekten mit - AWS SDKs für Java, .NET und PHP finden Sie unter Herunterladen von Objekten.

Beispiele für das Auflisten der Version von Objekten mit - AWS SDKs für .NET und Rust finden Sie unter Auflisten der Version von Objekten in einem Amazon S3-Bucket.

Verwenden der REST-API

Abrufen einer spezifischen Objektversion

1. Setzen Sie den Parameter `versionId` auf die ID der Version des Objekts, die Sie abrufen wollen.
2. Senden Sie eine `GET Object versionId`-Anforderung.

Example – Abrufen eines versionierten Objekts

Die folgende Anforderung ruft die Version `L4kqtJlcpXroDTDmpUMLUo` von ab `my-image.jpg`.

```
GET /my-image.jpg?versionId=L4kqtJlcpXroDTDmpUMLUo HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Sie können nur die Metadaten eines Objekts (nicht den Inhalt) abrufen. Weitere Informationen finden Sie unter the section called “Abrufen von Versionsmetadaten”.

Informationen zum Wiederherstellen einer früheren Objektversion finden Sie unter the section called “Wiederherstellen früherer Versionen”.

Abrufen der Metadaten einer Objektversion

Wenn Sie nur die Metadaten eines Objekts abrufen wollen (nicht seinen Inhalt), verwenden Sie die HEAD-Operation. Standardmäßig erhalten Sie die Metadaten der aktuellsten Version. Um die Metadaten einer spezifischen Objektversion abzurufen, müssen Sie ihre Versions-ID angeben.

Abruf der Metadaten einer Objektversion

1. Setzen Sie den Parameter `versionId` auf die ID der Version des Objekts, dessen Metadaten Sie abrufen wollen.
2. Senden Sie eine `HEAD Object versionId`-Anforderung.

Example – Abrufen der Metadaten eines versionierten Objekts

Die folgende Anforderung ruft die Metadaten der Version `3HL4kqCxf3vjVBH40Nrjfkd` von `my-image.jpg` ab.

```
HEAD /my-image.jpg?versionId=3HL4kqCxf3vjVBH40Nrjfkd HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Nachfolgend ist eine Beispielantwort angezeigt.

```
HTTP/1.1 200 OK
x-amz-id-2: ef8yU9AS1ed4OpIszj7UDNEHGran
x-amz-request-id: 318BC8BC143432E5
x-amz-version-id: 3HL4kqtJlcpXroDTDmjVBH40Nrjfkd
Date: Wed, 28 Oct 2009 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2006 12:00:00 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 434234
Content-Type: text/plain
Connection: close
Server: AmazonS3
```

Wiederherstellen früherer Versionen

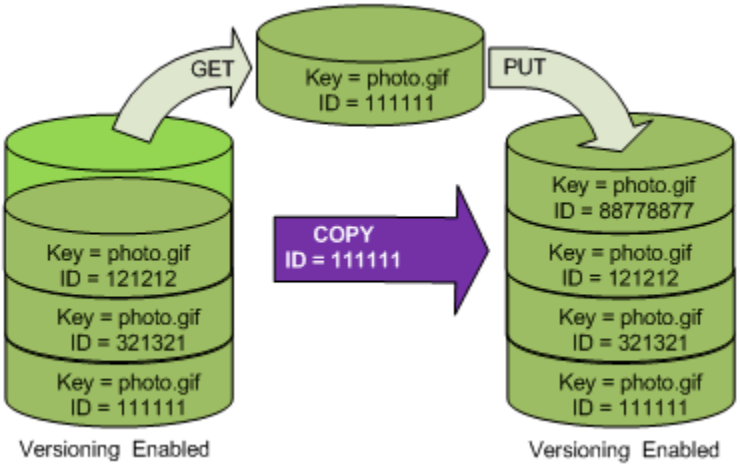
Sie können Versioning verwenden, um frühere Versionen eines Objekts abzurufen. Hierfür gibt es zwei Ansätze:

- Kopieren einer vorhergehenden Version des Objekts in denselben Bucket.

  Das kopierte Objekt wird zur aktuellen Version dieses Objekts, und alle Objektversionen werden beibehalten.
- Dauerhaftes Löschen der aktuellen Version des Objekts

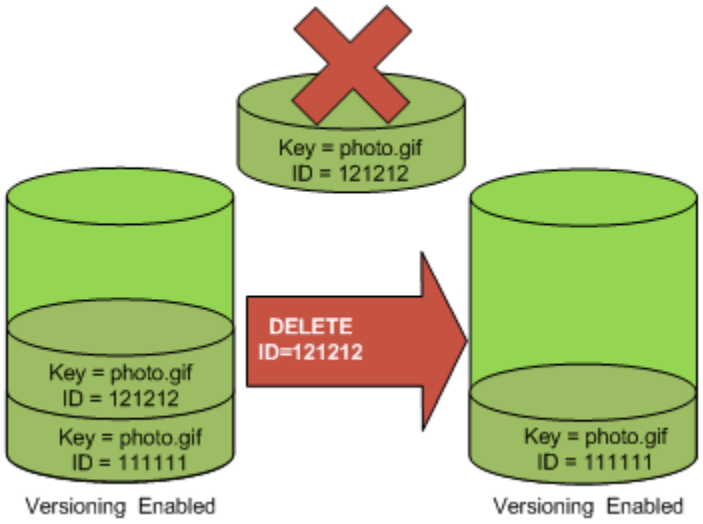
  Wenn Sie die aktuelle Objektversion löschen, wandeln Sie letztlich die nicht vorherige Version in die aktuelle Version dieses Objekts um.

Alle Objektversionen werden aufbewahrt, deshalb können Sie eine frühere Version zur aktuellen Version machen, indem Sie eine spezifische Version des Objekts in denselben Bucket kopieren. In der folgenden Abbildung wird das Quellobjekt (ID = 111111) in denselben Bucket kopiert. Amazon S3 stellt eine neue ID (88778877) bereit und diese wird zur aktuellen Version des Objekts. Der Bucket enthält jetzt also die ursprüngliche Objektversion (111111) und kopiert seine Kopie (88778877). Weitere Informationen zum Abrufen einer früheren Version und zum anschließenden Hochladen, um sie zur aktuellen Version zu machen, finden Sie unter Objektversionen aus einem versionsfähigen Bucket abrufen und Objekte hochladen.



Eine nachfolgende GET ruft Version 88778877 ab.

Die folgende Abbildung zeigt, wie die aktuelle Version (121212) eines Objekts gelöscht wird, sodass die vorhergehende Version (111111) zum aktuellen Objekt wird. Weitere Informationen zum Löschen eines Objekts finden Sie unter Löschen eines einzelnen Objekts.



Ein weiterer GET ruft die Version 111111 ab.

> **Note**
>
> Wenn Sie Objektversionen in Batches wiederherstellen möchten, können Sie die `CopyObject`-Operation verwenden. Die `CopyObject`-Operation kopiert jedes im Manifest angegebene Objekt. Objekte werden jedoch nicht zwingend in derselben Reihenfolge kopiert, in der sie im Manifest erscheinen. Wenn für versionierte Buckets die Beibehaltung der aktuellen/nicht aktuellen Versionsreihenfolge wichtig ist, sollten Sie zuerst alle nicht aktuellen Versionen kopieren. Kopieren Sie dann, nachdem der erste Auftrag abgeschlossen ist, die aktuellen Versionen in einen nachfolgenden Auftrag.

So stellen Sie frühere Objektversionen wieder her

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
3. Wählen Sie in der Liste Objects (Objekte) den Namen des Objekts aus.
4. Wählen Sie Versions (Versionen).

   Amazon S3 zeigt alle Versionen für das Objekt an.

5. Aktivieren Sie das Kontrollkästchen neben der Version ID (Versions-ID) für die Versionen, die Sie abrufen möchten.
6. Wählen Sie Actions (Aktionen), wählen Sie Download (Herunterladen) und speichern Sie das Objekt.

Objektversionen können auch in der Objektübersicht angezeigt, heruntergeladen und gelöscht werden. Weitere Informationen finden Sie unter Anzeigen einer Objektübersicht in der Amazon-S3-Konsole.

> **Important**
>
> Sie können den Löschvorgang für ein Objekt nur rückgängig machen, wenn seine aktuelle Version gelöscht wurde. Es ist nicht möglich, das Löschen einer vorherigen Version eines Objekts rückgängig zu machen, das gelöscht wurde. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Verwenden der AWS SDKs

Informationen zur Verwendung anderer AWS SDKs finden Sie im AWS -Entwicklerzentrum.

Python

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwendung der AWS SDK for Python (Boto).

Im folgenden Python-Codebeispiel wird die vorherige Version eines versionierten Objekts wiederhergestellt, indem alle Versionen gelöscht werden, die nach der angegebenen Rollback-Version aufgetreten sind.

```
def rollback_object(bucket, object_key, version_id):
    """
    Rolls back an object to an earlier version by deleting all versions that
    occurred after the specified rollback version.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that holds the object to roll back.
    :param object_key: The object to roll back.
```

```
    :param version_id: The version ID to roll back to.
    """
    # Versions must be sorted by last_modified date because delete markers are
    # at the end of the list even when they are interspersed in time.
    versions = sorted(
        bucket.object_versions.filter(Prefix=object_key),
        key=attrgetter("last_modified"),
        reverse=True,
    )

    logger.debug(
        "Got versions:\n%s",
        "\n".join(
            [
                f"\t{version.version_id}, last modified {version.last_modified}"
                for version in versions
            ]
        ),
    )

    if version_id in [ver.version_id for ver in versions]:
        print(f"Rolling back to version {version_id}")
        for version in versions:
            if version.version_id != version_id:
                version.delete()
                print(f"Deleted version {version.version_id}")
            else:
                break

        print(f"Active version is now {bucket.Object(object_key).version_id}")
    else:
        raise KeyError(
            f"{version_id} was not found in the list of versions for "
f"{object_key}."
        )
```

Löschen von Objekten aus einem versioning-fähigen Bucket

Sie können Objektversionen aus Amazon-S3-Buckets löschen, wann immer Sie möchten. Sie können auch Lebenszyklus-Konfigurationsregeln für Objekte definieren, die einen definierten Lebenszyklus

haben, um Amazon S3 aufzufordern, die aktuellen Objektversionen ablaufen zu lassen, oder die nicht aktuellen Objektversionen ständig zu entfernen. Wenn Ihr Bucket versioning-fähig ist oder das Versioning ausgesetzt ist, funktionieren die Lebenszyklus-Konfigurationsaktionen wie folgt:

- Die Aktion `Expiration` gilt für die aktuelle Objektversion. Statt die aktuelle Objektversion zu löschen, behält Amazon S3 die aktuelle Version als nicht aktuelle Version bei, indem es eine Löschmarkierung hinzufügt, die anschließend zur aktuellen Version wird.
- Die `NoncurrentVersionExpiration`-Aktion wird nur auf nicht aktuelle Objektversionen angewendet, und Amazon S3 entfernt diese Objektversionen dauerhaft. Dauerhaft entfernte Objekte können nicht wiederhergestellt werden.

Weitere Informationen zu S3 Lifecycle finden Sie unter Verwalten Ihres Speicher-Lebenszyklus und Beispiele der S3-Lebenszyklus-Konfiguration.

Um zu sehen, wie viele aktuelle und nicht aktuelle Objektversionen Ihre Buckets enthalten, können Sie die Metriken von Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Weitere Informationen finden Sie unter Verwenden von S3 Storage Lens zur Optimierung Ihrer Speicherkosten. Eine vollständige Liste der Metriken finden Sie im Glossar der S3-Storage-Lens-Metriken.

> **Note**
>
> Für jede Version eines Objekts, das gespeichert und übertragen wird, einschließlich nicht aktueller Objektversionen, gelten die normalen Amazon S3-Tarife. Weitere Informationen finden Sie unter Amazon S3 – Preise.

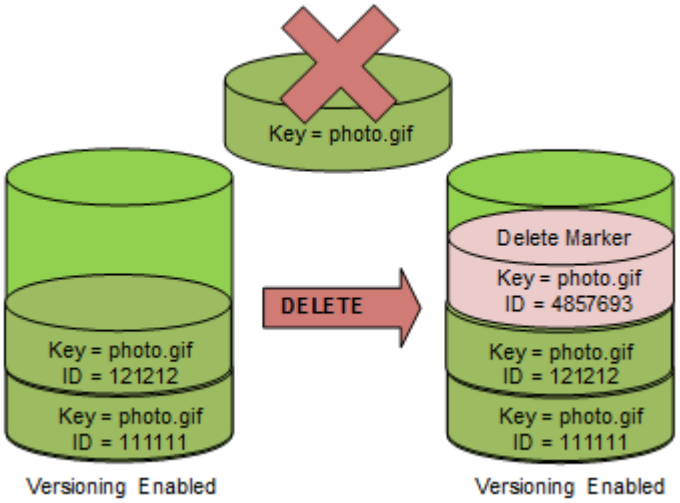
Löschen von Anwendungsfällen

Eine `DELETE`-Anforderung hat die folgenden Anwendungsfälle:

- Wenn das Versioning aktiviert ist, kann ein einfaches `DELETE` ein Objekt nicht dauerhaft löschen. (Eine einfache `DELETE`-Anforderung ist eine Anforderung, die keine Versions-ID angibt.) Stattdessen fügt Amazon S3 eine Löschmarkierung in den Bucket ein, und die Löschmarkierung wird zur aktuellen Objektversion mit einer neuen ID.

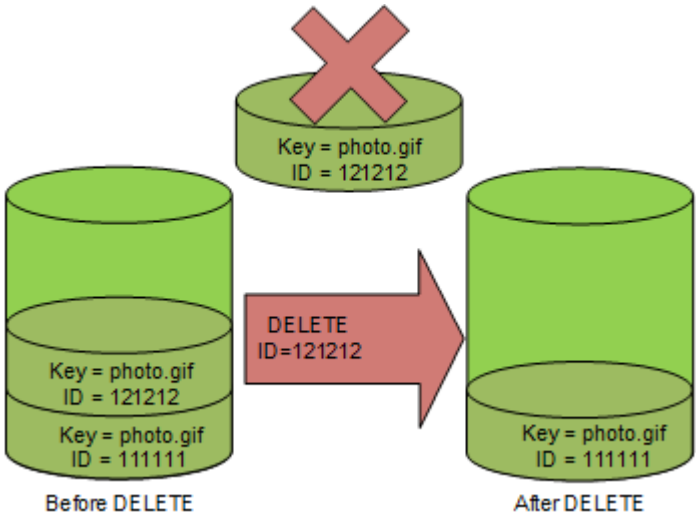
Wenn Sie versuchen, `GET` für ein Objekt auszuführen, dessen aktuelle Version eine Löschmarkierung ist, verhält sich Amazon S3, als wäre das Objekt gelöscht worden (obwohl es nicht vollständig entfernt wurde), und gibt einen 404-Fehler zurück. Weitere Informationen finden Sie unter Arbeiten mit Löschmarkierungen.

Die folgende Abbildung zeigt, dass ein einfaches `DELETE` das spezifizierte Objekt nicht wirklich löscht. Stattdessen fügt Amazon S3 eine Löschmarkierung ein.



- Um Objektversionen dauerhaft zu löschen, müssen Sie verwende `DELETE Object versionId`.

  Die folgende Abbildung zeigt, dass das beim Löschen einer angegebenen Objektversion dieses Objekt dauerhaft gelöscht wird.



Löschen von Objektversionen

Sie können Objektversionen in Amazon S3 mithilfe der Konsole, AWS SDKs, der REST-API oder der löschen AWS Command Line Interface.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
3. Wählen Sie in der Liste Objects (Objekte) den Namen des Objekts aus.
4. Wählen Sie Versions (Versionen).

   Amazon S3 zeigt alle Versionen für das Objekt an.
5. Aktivieren Sie das Kontrollkästchen neben der Version ID (Versions-ID) für die Versionen, die Sie permanent löschen möchten.
6. Wählen Sie Delete (Löschen).
7. Geben Sie in Objekte endgültig löschen? **`permanently delete`** ein.

   > ⚠ Warning
   >
   > Wenn Sie eine Objektversion endgültig löschen, kann die Aktion nicht rückgängig gemacht werden.
8. Wählen Sie Delete objects (Objekte löschen).

   Amazon S3 löscht die Objektversion.

Verwenden der AWS SDKs

Beispiele für das Löschen von Objekten mit den - AWS SDKs für Java, .NET und PHP finden Sie unter Löschen von Amazon-S3-Objekten. Die Beispiele für das Löschen von Objekten in nicht versioning-fähigen und versioning-fähigen Buckets sind dieselben. Für versioning-fähige Buckets weist Amazon S3 jedoch eine Versionsnummer zu. Andernfalls ist die Versionsnummer null.

Informationen zur Verwendung anderer AWS SDKs finden Sie im AWS -Entwicklerzentrum.

Python

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwendung der AWS SDK for Python (Boto).

Das folgende Beispiel für Python-Code zeigt das dauerhafte Löschen eines versionierten Objekts, indem alle seine Versionen gelöscht werden.

```
def permanently_delete_object(bucket, object_key):
    """
    Permanently deletes a versioned object by deleting all of its versions.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to delete.
    """
    try:
        bucket.object_versions.filter(Prefix=object_key).delete()
        logger.info("Permanently deleted all versions of object %s.", object_key)
    except ClientError:
        logger.exception("Couldn't delete all versions of %s.", object_key)
        raise
```

Verwenden der REST-API

Löschen einer spezifischen Objektversion

- Geben Sie in einem `DELETE` eine Versions-ID ein.

Example – Löschen einer spezifischen Version

Im folgenden Beispiel wird die Version `UIORUnfnd89493jJFJ` von `photo.gif` gelöscht.

```
DELETE /photo.gif?versionId=UIORUnfnd89493jJFJ HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 12 Oct 2009 17:50:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:xQE0diMbLRepdf3YB+FIEXAMPLE=
```

```
Content-Type: text/plain
Content-Length: 0
```

Verwenden der AWS CLI

Der folgende Befehl löscht ein Objekt namens test.txt aus einem Bucket mit dem Namen *DOC-EXAMPLE-BUCKET1*. Zum Entfernen einer bestimmte Version eines Objekts müssen Sie der Bucket-Eigentümer sein und die Versions-ID-Subressource verwenden.

```
aws s3api delete-object --bucket DOC-EXAMPLE-BUCKET1 --key test.txt --version-
id versionID
```

Weitere Informationen zu `delete-object` finden Sie unter delete-object in der AWS CLI - Befehlsreferenz.

Weitere Informationen zum Löschen von Objektversionen finden Sie in den folgenden Themen:

- Arbeiten mit Löschmarkierungen
- Entfernen von Löschmarkierungen, um eine ältere Version aktuell zu machen
- Löschen eines Objekts aus einem MFA-Delete-fähigen Bucket

Arbeiten mit Löschmarkierungen

Eine Löschmarkierung in Amazon S3 ist ein Platzhalter (bzw. eine Markierung) für ein versioniertes Objekt, das in einer einfachen `DELETE`-Anforderung angegeben wurde. Eine einfache `DELETE`-Anforderung ist eine Anforderung, die keine Versions-ID angibt. Weil sich das Objekt in einem versioning-fähigen Bucket befindet, wurde das Objekt nicht gelöscht. Die Löschmarkierung bewirkt jedoch, dass sich Amazon S3 verhält, als wäre das Objekt gelöscht worden. Sie können einen `DELETE`-API-Aufruf in Amazon S3 für eine Löschmarkierung verwenden. Dazu müssen Sie die `DELETE` Anforderung unter Verwendung eines AWS Identity and Access Management (IAM)-Benutzers oder einer Rolle mit den entsprechenden Berechtigungen stellen.

Eine Löschmarkierung hat einen Schlüsselnamen (oder Schlüssel) und eine Versions-ID, so wie jedes andere Objekt. Eine Löschmarkierung unterscheidet sich jedoch wie folgt von anderen Objekten:

- Einer Löschmarkierung sind keine Daten zugeordnet.
- Eine Löschmarkierung ist nicht mit einem Wert für eine Zugriffssteuerungsliste (ACL) verknüpft.

- Wenn Sie eine `GET`-Anforderung für eine Löschmarkierung ausgeben, ruft die `GET`-Anforderung nichts ab, da eine Löschmarkierung keine Daten enthält. Insbesondere, wenn in Ihrer `GET`-Anforderung keine `versionId` angegeben ist, erhalten Sie den Fehler 404 (Nicht gefunden).

Für Löschmarkierungen fällt eine Mindestgebühr für die Speicherung in Amazon S3 an.
Der Speicherbedarf einer Löschmarkierung ist gleich der Größe des Schlüsselnamens der Löschmarkierung. Ein Schlüsselname ist eine Folge von Unicode-Zeichen. Die UTF-8-Codierung für den Schlüsselnamen fügt Ihrem Bucket für jedes Zeichen im Namen 1 bis 4 Byte Speicherplatz hinzu. Löschmarkierungen werden in der S3-Standardspeicherklasse gespeichert.

Wenn Sie ermitteln möchten, wie viele Löschmarkierungen Sie haben und in welcher Speicherklasse diese gespeichert sind, können Sie Amazon S3 Storage Lens verwenden. Weitere Informationen finden Sie unter Bewerten Ihrer Speicheraktivität und -nutzung mit Amazon S3 Storage Lens und Amazon S3-Storage-Lens-Metrikglossar.

Weitere Informationen zu Schlüsselnamen finden Sie unter Erstellen von Objektschlüsselnamen. Informationen zum Löschen von Löschmarkierungen finden Sie unter Verwalten von Löschmarkierungen.

Nur Amazon S3 kann eine Löschmarkierung erstellen. Dies erfolgt immer, wenn Sie eine `DeleteObject`-Anfrage für ein Objekt in einem Bucket mit Versioning oder ausgesetztem Versioning stellen. Das in der `DELETE`-Anforderung angegebene Objekt wird nicht wirklich gelöscht. Stattdessen wird die Löschmarkierung zur aktuellen Version des Objekts. Der Schlüsselname (oder Schlüssel) des Objekts wird zum Schlüssel der Löschmarkierung.

Wenn Sie ein Objekt abrufen, ohne eine `versionId` in der Anforderung anzugeben, und die aktuelle Version eine Löschmarkierung ist, antwortet Amazon S3 mit Folgendem:

- Fehler 404 (Nicht gefunden)
- Ein Antwort-Header `x-amz-delete-marker: true`

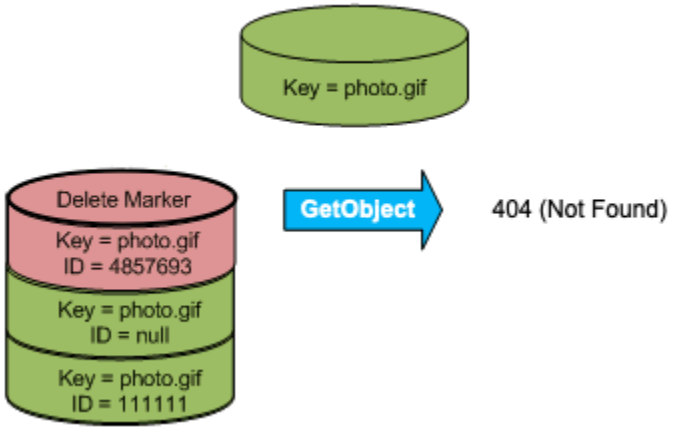
Wenn Sie ein Objekt abrufen, indem Sie eine `versionId` in der Anforderung angeben, und die angegebene Version eine Löschmarkierung ist, antwortet Amazon S3 mit Folgendem:

- Ein Fehler 405 (Method Not Allowed)
- Ein Antwort-Header `x-amz-delete-marker: true`
- Ein Antwort-Header, `Last-Modified: timestamp` (nur bei Verwendung der - HeadObject oder GetObject-API-Operationen)

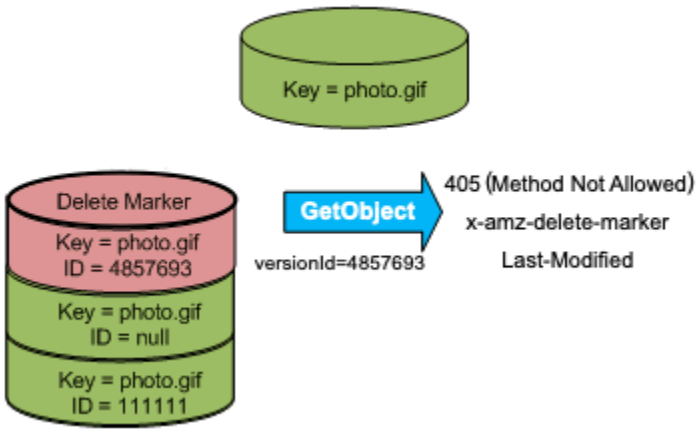
Der Antwort-Header `x-amz-delete-marker: true` teilt Ihnen mit, dass das Objekt, auf das Sie zugegriffen haben, eine Löschmarkierung war. Dieser Antwort-Header gibt niemals `false` zurück, denn wenn der Wert `false` lautet, handelt es sich bei der aktuellen oder angegebenen Version des Objekts nicht um eine Löschmarkierung.

Der Antwort-Header `Last-Modified` gibt die Erstellungszeit der Löschmarkierungen an.

Die folgende Abbildung zeigt, wie ein `GetObject`-API-Aufruf für ein Objekt, dessen aktuelle Version eine Löschmarkierung ist, mit dem Fehler 404 (Nicht gefunden) antwortet. Der Antwort-Header enthält `x-amz-delete-marker: true`.

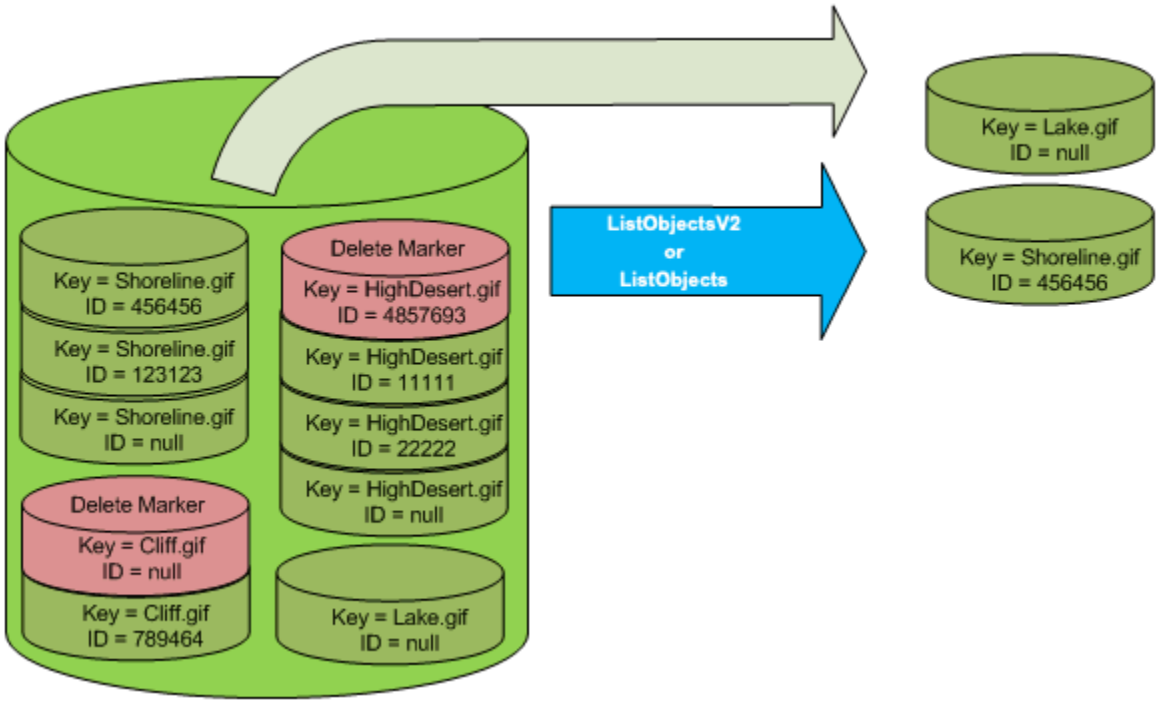


Wenn Sie einen `GetObject`-Aufruf für ein Objekt durchführen, indem Sie eine `versionId` in Ihrer Anfrage angeben, und wenn die angegebene Version eine Löschmarkierung ist, antwortet Amazon S3 mit dem Fehler 405 (Methode nicht zulässig) und die Antwort-Header enthalten `x-amz-delete-marker: true` und `Last-Modified: timestamp`.



Die einzige Methode, Löschmarkierungen (und andere Versionen eines Objekts) aufzulisten, ist die Verwendung der `versions`-Subressource in einer ListObjectVersions-Anforderung. Die folgende

Abbildung zeigt, dass eine ListObjectsV2- oder ListObjects-Anforderung keine Objekte zurückgibt, deren aktuelle Version eine Löschmarkierung ist.



Verwalten von Löschmarkierungen

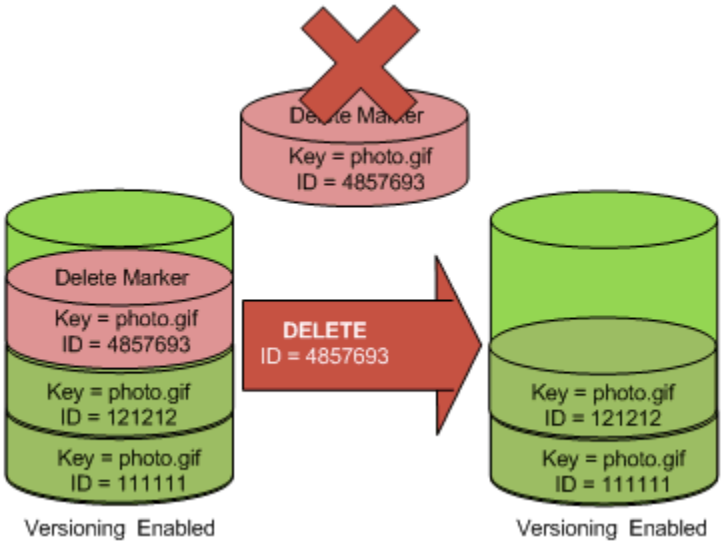
Konfigurieren des Lebenszyklus zum automatischen Bereinigen abgelaufener Löschmarkierungen

Eine abgelaufene Objektlöschmarkierung ist eine, bei der alle Objektversionen gelöscht werden und nur eine einzelne Löschmarkierung erhalten bleibt. Wenn die Lebenszyklus-Konfiguration so eingerichtet ist, dass aktuelle Versionen gelöscht werden, oder die `ExpiredObjectDeleteMarker`-Aktion explizit festgelegt ist, entfernt Amazon S3 die Löschmarkierung des abgelaufenen Objekts. Ein Beispiel finden Sie unter Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte.

Entfernen von Löschmarkierungen, um eine ältere Version aktuell zu machen

Wenn Sie ein Objekt in einem für das Versioning geeignetes Bucket löschen, bleiben alle Versionen in dem Bucket und Amazon S3 erstellt eine Löschmarkierung für das Objekt. Um das Löschen des Objekts rückgängig zu machen, müssen Sie diese Löschmarkierung löschen. Weitere Informationen zum Versioning und zu Löschmarkierungen finden Sie unter Verwenden der Versioning in S3-Buckets.

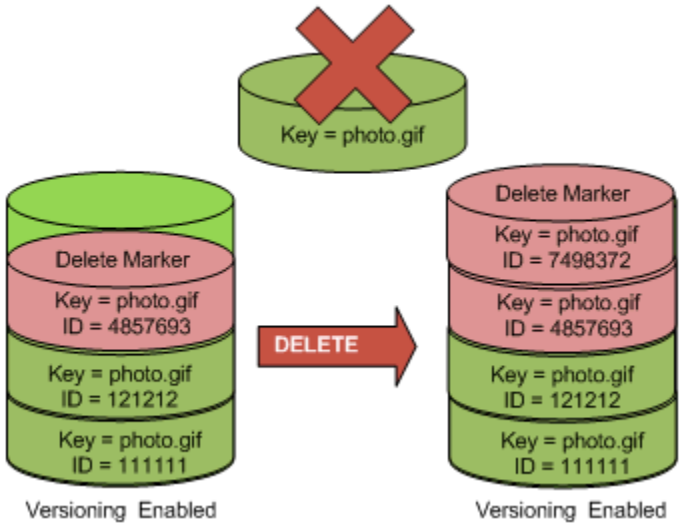
Um eine Löschmarkierung dauerhaft zu löschen, müssen Sie Ihre Versions-ID in einer `DeleteObject versionId`-Anforderung angeben. Die folgende Abbildung zeigt, dass ein einfaches `DeleteObject versionId` eine Löschmarkierung nicht dauerhaft entfernt.



Das Entfernen der Löschmarkierung bewirkt, dass eine einfache `GET`-Anforderung jetzt die aktuelle Versions-ID (121212) des Objekts abruft.

> **Note**
>
> Wenn Sie eine `DeleteObject`-Anforderung verwenden, bei der die aktuelle Version eine Löschmarkierung ist (ohne die Versions-ID der Löschmarkierung anzugeben), löscht Amazon S3 die Löschmarkierung nicht, sondern eine andere `PUT`s Löschmarkierung.

Um eine Löschmarkierung mit einer `NULL`-Versions-ID zu löschen, müssen Sie das `NULL` als Versions-ID in der `DeleteObject`-Anforderung übergeben. Die folgende Abbildung zeigt, wie eine einfache `DeleteObject`-Anfrage ohne Versions-ID, bei der die aktuelle Version eine Löschmarkierung ist, nichts entfernt, sondern stattdessen eine zusätzliche Löschmarkierung mit einer eindeutigen Versions-ID (7498372) hinzufügt.



Verwenden der S3-Konsole

Mit den folgenden Schritten können Sie gelöschte Objekte wiederherstellen, die keine Ordner aus Ihrem S3-Bucket sind, einschließlich der Objekte, die sich in diesen Ordnern befinden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des von Ihnen erstellten Buckets aus.
3. Um eine Liste der Versionen der Objekte in dem Bucket anzuzeigen, wählen Sie den Schalter List versions (Versionen auflisten). Sie sehen die Löschmarkierungen für gelöschte Objekte.
4. Um das Löschen eines Objekts rückgängig zu machen, müssen Sie die Löschmarkierung löschen. Markieren Sie das Kontrollkästchen neben der Löschmarkierung des Objekts, das wiederhergestellt werden soll, und wählen Sie dann Delete (Löschen).
5. Bestätigen Sie den Löschvorgang auf der Seite Delete objects (Objekte löschen) .
   a. Geben Sie für Permanently delete objects? (Objekte dauerhaft löschen?) **`permanently delete`** ein.
   b. Wählen Sie Delete objects (Objekte löschen).

> **Note**
>
> Sie können die Amazon-S3-Konsole nicht verwenden, um das Löschen von Ordnern rückgängig zu machen. Sie müssen die AWS CLI oder das SDK verwenden. Beispiele finden Sie unter How can I retrieve an Amazon S3 object that was deleted in a versioning-enabled

bucket? (Wie kann ich ein Amazon-S3-Objekt wiederherstellen, das in einem Versioning-fähigen Bucket gelöscht wurde?) im AWS -Wissenszentrum

Verwenden der REST-API

Eine Löschmarkierung dauerhaft entfernen

1. Setzen Sie den Parameter `versionId` auf die ID der Version der Löschmarkierung, die Sie entfernen wollen.
2. Senden Sie eine `DELETE Object versionId`-Anforderung.

Example – Entfernen einer Löschmarkierung

Das folgende Beispiel entfernt die Löschmarkierung für `photo.gif` Version 4857693.

```
DELETE /photo.gif?versionId=4857693 HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Wenn Sie eine Löschmarkierung löschen, nimmt Amazon S3 folgendes in die Antwort auf.

```
204 NoContent
x-amz-version-id: versionID
x-amz-delete-marker: true
```

Verwenden der AWS SDKs

Informationen zur Verwendung anderer AWS SDKs finden Sie im AWS-Entwicklerzentrum.

Python

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwendung der AWS SDK for Python (Boto).

Das folgende Python-Codebeispiel zeigt, wie Sie einen Löschmarker von einem Objekt entfernen und damit die neueste nicht-aktuelle Version der aktuellsten Version zur aktuellen Version des Objekts machen.

```
def revive_object(bucket, object_key):
```

```
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
marker.
    By removing the delete marker, we make the previous version the latest version
    and the object then presents as *not* deleted.

    Usage is shown in the usage_demo_single_object function at the end of this
module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to revive.
    """
    # Get the latest version for the object.
    response = s3.meta.client.list_object_versions(
        Bucket=bucket.name, Prefix=object_key, MaxKeys=1
    )

    if "DeleteMarkers" in response:
        latest_version = response["DeleteMarkers"][0]
        if latest_version["IsLatest"]:
            logger.info(
                "Object %s was indeed deleted on %s. Let's revive it.",
                object_key,
                latest_version["LastModified"],
            )
            obj = bucket.Object(object_key)
            obj.Version(latest_version["VersionId"]).delete()
            logger.info(
                "Revived %s, active version is now %s  with body '%s'",
                object_key,
                obj.version_id,
                obj.get()["Body"].read(),
            )
        else:
            logger.warning(
                "Delete marker is not the latest version for %s!", object_key
            )
    elif "Versions" in response:
        logger.warning("Got an active version for %s, nothing to do.", object_key)
    else:
        logger.error("Couldn't get any version info for %s.", object_key)
```

Löschen eines Objekts aus einem MFA-Delete-fähigen Bucket

Wenn die Versioning-Konfiguration MFA Delete unterstützt, muss der Bucket-Eigentümer den `x-amz-mfa`-Anfrage-Header in Anfragen aufnehmen, um eine Objektversion dauerhaft zu löschen oder den Versioning-Status des Buckets zu ändern. Anforderung mit `x-amz-mfa` müssen HTTPS verwenden.

Der Wert des Headers ist die Verkettung der Seriennummer Ihres Authentifizierungsgeräts, eines Leerzeichens und des darauf angezeigten Authentifizierungscodes. Wenn sie dies im Anforderungs-Header nicht angeben, schlägt die Anforderung fehl.

Weitere Informationen zu Authentifizierungsgeräten finden Sie unter Multi-Factor Authentication.

Example – Löschen eines Objekts aus einem MFA Delete-fähigen Bucket

Im folgenden Beispiel wird `my-image.jpg` (in der angegebenen Version) gelöscht, ein mit MFA Delete konfigurierter Bucket.

Beachten Sie das Leerzeichen zwischen *`[SerialNumber]`* und *`[AuthenticationCode].`* Weitere Informationen finden Sie unter DeleteObject in der API-Referenz zu Amazon Simple Storage Service.

```
DELETE /my-image.jpg?versionId=3HL4kqCxf3vjVBH40Nrjfkd HTTPS/1.1
Host: bucketName.s3.amazonaws.com
x-amz-mfa: 20899872 301749
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Weitere Informationen zur Aktivierung von MFA Delete finden Sie unter Konfigurieren von MFA Delete.

Konfigurieren von versionierten Objektberechtigungen

Die Berechtigungen für Objekte in Amazon S3 werden auf Versionsebene festgelegt. Jede Version hat ihren eigenen Objekteigentümer. Das AWS-Konto , das die Objektversion erstellt, ist der Eigentümer. Sie können also unterschiedliche Berechtigungen für unterschiedliche Versionen desselben Objekts einrichten. Dazu geben Sie die Versions-ID des Objekts an, dessen

Berechtigungen Sie in einer `PUT Object versionId acl`-Anforderung setzen wollen. Eine detaillierte Beschreibung sowie Anweisungen zur Verwendung von ACLs finden Sie unter Identity and Access Management in Amazon S3

Example – Einrichtung von Berechtigungen für eine Objektversion

Die folgende Anforderung legt die Berechtigung für den Empfänger, `BucketOwner@amazon.com`, auf `FULL_CONTROL` für den Schlüssel, `my-image.jpg`, Versions-ID, `3HL4kqtJvjVBH40Nrjfkd` fest.

```
PUT /my-image.jpg?acl&versionId=3HL4kqtJvjVBH40Nrjfkd HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
Content-Length: 124

<AccessControlPolicy>
  <Owner>
    <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
    <DisplayName>mtd@amazon.com</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>a9a7b886d6fd24a52fe8ca5bef65f89a64e0193f23000e241bf9b1c61be666e9</ID>
        <DisplayName>BucketOwner@amazon.com</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
  </AccessControlPolicy>
```

Analog dazu müssen Sie, um die Berechtigungen für eine spezifische Objektversion zu erhalten, ihre Versions-ID in einer `GET Object versionId acl`-Anforderung angeben. Sie müssen diese Versions-ID angeben, weil `GET Object acl` standardmäßig die Berechtigungen für die aktuelle Version des Objekts zurückgibt.

Example – Abrufen der Berechtigungen für eine bestimmte Objektversion

Im folgenden Beispiel gibt Amazon S3 die Berechtigungen für den Schlüssel, `my-image.jpg`, Versions-ID, `DVBH40Nr8X8gUMLUo` zurück.

```
GET /my-image.jpg?versionId=DVBH40Nr8X8gUMLUo&acl HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU
```

Weitere Informationen finden Sie unter GetObjectAcl in der API-Referenz zu Amazon Simple Storage Service.

Arbeiten mit Objekten in einem Bucket mit ausgesetztem Versioning

In Amazon S3 können Sie das Versioning aussetzen, um zu verhindern, dass sich neue Versionen desselben Objekts in einem Bucket ansammeln. Sie könnten dies tun, weil Sie nur eine einzige Version eines Objekts in einem Bucket haben möchten. Oder Sie möchten möglicherweise keine Gebühren für mehrere Versionen anfallen.

Wenn Sie das Versioning aussetzen, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anfragen verarbeitet. Die Themen in diesem Abschnitt erläutern verschiedene Objekt-Vorgänge in einem Bucket mit ausgesetztem Versioning, einschließlich Hinzufügen, Abrufen und Löschen von Objekten.

Weitere Informationen über das S3-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets. Weitere Informationen über das Abrufen von Objektversionen, finden Sie unter Abrufen von Objektversionen aus einem versioning-fähigen Bucket.

Themen

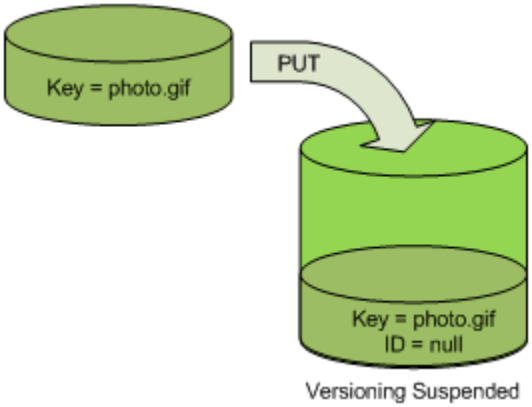
- Hinzufügen von Objekten zu Buckets mit ausgesetztem Versioning
- Abrufen von Objekten aus Buckets mit ausgesetztem Versioning
- Löschen von Objekten aus Buckets mit ausgesetztem Versioning

Hinzufügen von Objekten zu Buckets mit ausgesetztem Versioning

Sie können in Amazon S3 Objekte in Buckets mit ausgesetztem Versioning hinzufügen, um das Objekt mit der Versions-ID null zu erzeugen oder eine Objektversion mit einer übereinstimmenden Versions-ID zu überschreiben.

Nachdem Sie das Versioning für einen Bucket ausgesetzt haben, fügt Amazon S3 automatisch jedem nachfolgend (mit `PUT`, `POST` oder `CopyObject`) in diesem Bucket gespeicherten Objekt die Versions-ID `null` hinzu

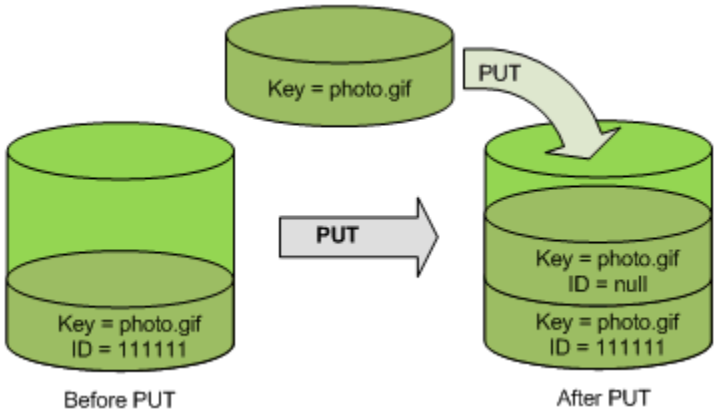
Die folgende Abbildung zeigt, wie Amazon S3 jedem Objekt automatisch die Versions-ID `null` hinzufügt, wenn es einem Bucket mit ausgesetztem Versioning hinzugefügt wird.



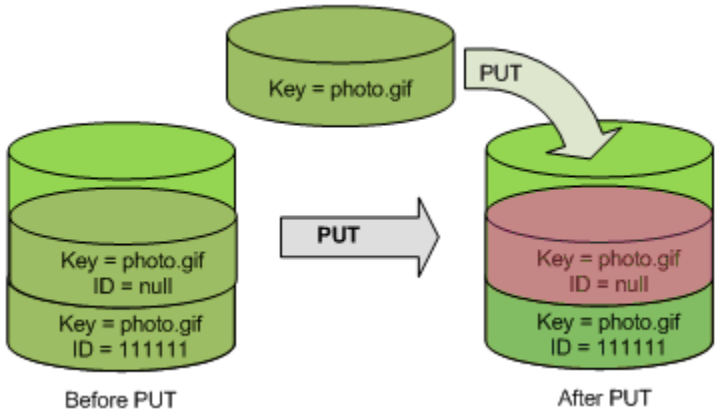
Wenn sich bereits eine Null-Version im Bucket befindet und Sie ein weiteres Objekt mit demselben Schlüssel hinzufügen, überschreibt das hinzugefügte Objekt die ursprüngliche Null-Version.

Wenn es versionsfähige Objekte im Bucket gibt, wird die Version, die Sie mit PUT hinzufügen, zur aktuellen Version des Objekts. Die folgende Abbildung zeigt, wie das Hinzufügen eines Objekts in einen Bucket, der versionsfähige Objekte enthält, das bereits im Bucket enthaltene Objekt nicht überschreibt.

In diesem Fall befand sich Version 111111 bereits im Bucket. Amazon S3 weist dem Objekt, das hinzugefügt und im Bucket gespeichert wird, die Versions-ID null zu. Version 111111 wird nicht überschrieben.



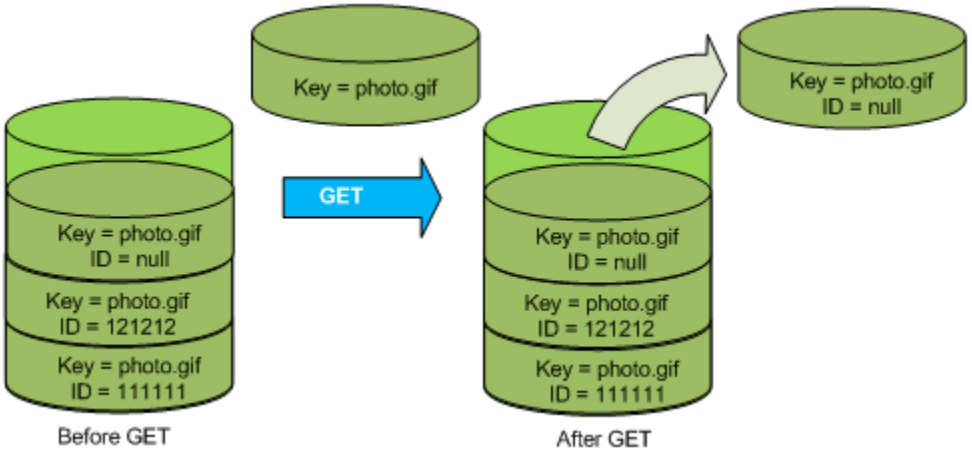
Wenn bereits eine Null-Version in einem Bucket vorhanden ist, wird die Null-Version überschrieben, wie in der folgenden Abbildung gezeigt.



Obwohl der Schlüssel und die Versions-ID (`null`) der Null-Version vor und nach PUT gleich sind, wird der Inhalt der ursprünglich im Bucket gespeicherten Version jedoch durch die Inhalte des Objekts ersetzt, die mit PUT in den Bucket geschrieben wurde.

Abrufen von Objekten aus Buckets mit ausgesetztem Versioning

Eine `GET Object`-Anforderung gibt die aktuelle Version eines Objekts zurück, unabhängig davon, ob Sie das Versioning für einen Bucket aktiviert haben oder nicht. Die folgende Abbildung zeigt, wie ein einfaches `GET` die aktuelle Version des Objekts zurückgibt.



Löschen von Objekten aus Buckets mit ausgesetztem Versioning

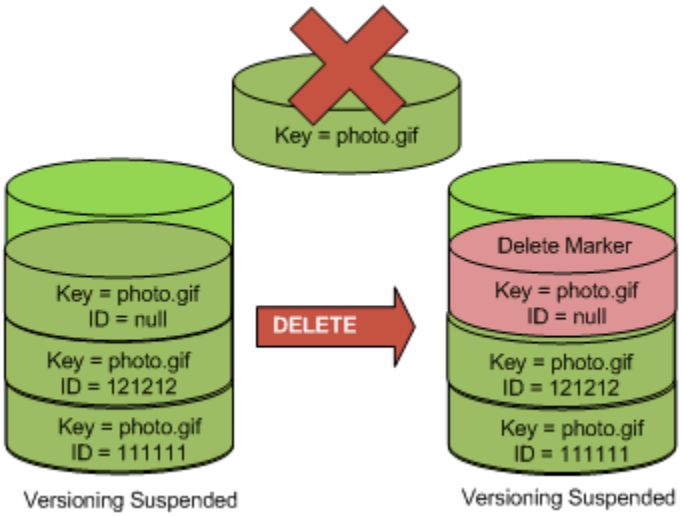
Sie können Objekte aus Buckets mit ausgesetztem Versioning löschen, um ein Objekt mit einer Versions-ID von null zu entfernen.

Wenn das Versioning für ein Bucket ausgesetzt ist, gilt für eine `DELETE`-Anforderung:

- Kann nur ein Objekt entfernen, dessen Versions-ID `null` ist.
- Entfernt nichts, wenn es keine Nullversion des Objekts im Bucket gibt.

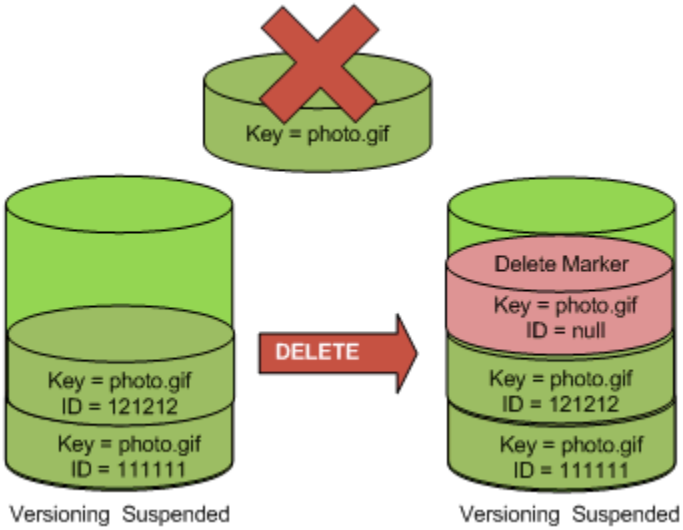
- Fügt eine Löschmarkierung in den Bucket ein.

Die folgende Abbildung zeigt, wie ein einfaches `DELETE` eine Null-Version entfernt. (Eine einfache `DELETE`-Anforderung ist eine Anforderung, die keine Versions-ID angibt.) Amazon S3 fügt an seiner Stelle eine Löschmarkierung mit einer Versions-ID von `null` ein.



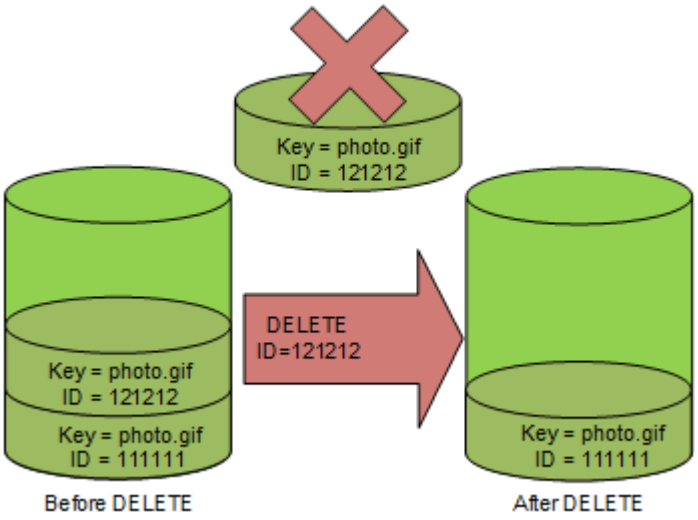
Beachten Sie, dass eine Löschmarkierung keinen Inhalt hat, Sie verlieren also den Inhalt der Nullversion, wenn sie durch eine Löschmarkierung ersetzt wird.

Die folgende Abbildung zeigt einen Bucket, der keine Nullversion enthält. In diesem Fall entfernt `DELETE` nichts, sondern Amazon S3 fügt einfach eine Löschmarkierung ein.



Selbst in einem Bucket mit ausgesetztem Versioning kann der Bucket-Eigentümer eine spezifische Version dauerhaft löschen, indem er die Versions-ID in der `DELETE`-Anfrage mit einbezieht. Die

folgende Abbildung zeigt, dass das beim Löschen einer angegebenen Objektversion diese Version des Objekts dauerhaft gelöscht wird. Nur der Bucket-Eigentümer kann eine spezifische Objektversion löschen.



Verwenden von AWS Backup für Amazon S3

Amazon S3 ist nativ integriert in AWS Backup, ein vollständig verwalteter, richtlinienbasierter Service, mit dem Sie Backup-Richtlinien zum Schutz Ihrer Amazon-S3-Daten zentral definieren können. Nachdem Sie Ihre Backup-Richtlinien definiert und ihnen Amazon-S3-Ressourcen zugewiesen haben, automatisiert AWS Backup die Erstellung von Amazon-S3-Backups und speichert sie sicher in einem verschlüsselten Sicherungstresor, den Sie in Ihrem Backup-Plan festlegen.

Sie können die folgenden Aktionen mit AWS Backup für Amazon S3 ausführen:

- Erstellen kontinuierlicher Sicherungen und regelmäßiger Sicherungen. Kontinuierliche Sicherungen sind nützlich für die Point-in-Time Wiederherstellung,und regelmäßige Backups eignen sich, um Ihre Bedürfnisse für die langfristige Datenaufbewahrung zu erfüllen.
- Automatisieren der Planung und Aufbewahrung von Backups durch die zentrale Konfigurierung von Backuprichtlinien.
- Wiederherstellen von Backups von Amazon-S3-Daten zu einem bestimmten Zeitpunkt.

Zusammen mit AWS Backup können Sie die S3-Versionsverwaltung und -Replikation verwenden, um versehentliche Löschungen wiederherzustellen und Ihre eigenen Selbst-Wiederherstellungs-Vorgänge durchzuführen.

Voraussetzungen

Sie müssen die S3-Versionsverwaltung in Ihrem Bucket aktivieren, bevor AWS Backup ein Backup erstellen kann.

> **Note**
>
> Wir empfehlen, dass Sie eine andere Lebenszyklusablaufregel für versionsfähige Buckets festlegen, von denen ein Backup erstellt wird. Wenn Sie keinen Lebenszyklusablaufzeitraum festlegen, können Ihre Amazon-S3-Speicherkosten steigen, weil AWS Backup alle Versionen Ihrer Amazon-S3-Daten beibehält.

Erste Schritte

Informationen zu den ersten Schritten mit AWS Backup für Amazon S3 finden Sie unter Erstellen von Amazon-S3-Backups im Entwicklerhandbuch für AWS Backup.

Beschränkungen und Einschränkungen

Weitere Informationen über die Einschränkungen finden Sie unter Erstellen von Amazon-S3-Backups im Entwicklerhandbuch für AWS Backup.

Arbeiten mit archivierten Objekten

Sie können Objekte, auf die selten zugegriffen wird, archivieren, um Ihre Speicherkosten dafür zu senken. Wenn Sie ein Objekt archivieren, wird es in einen kostengünstigen Speicher verschoben. Sie können also nicht in Echtzeit darauf zugreifen.

Auf archivierte Objekte kann zwar nicht in Echtzeit zugegriffen werden, Sie können sie jedoch je nach Speicherklasse innerhalb von Minuten oder Stunden wiederherstellen. Sie können ein archiviertes Objekt mithilfe der Amazon-S3-Konsole, S3 Batch Operations, der REST-API, den AWS SDKs und der AWS Command Line Interface (AWS CLI) wiederherstellen. Detaillierte Anweisungen finden Sie unter Wiederherstellen eines archivierten Objekts.

Amazon-S3-Objekte in den folgenden Speicherklassen oder -stufen werden archiviert. Es kann nicht in Echtzeit darauf zugegriffen werden:

- Die Speicherklasse S3 Glacier Flexible Retrieval
- Die Speicherklasse S3 Glacier Deep Archive

- Die Zugriffsebene S3 Intelligent-Tiering Archive
- Die S3-Intelligent-Tiering-Deep-Archive-Zugriffsebene

Um archivierte Objekte wiederherzustellen, müssen Sie die folgenden Schritte ausführen:

- Für Objekte in den Speicherklassen S3 Glacier Flexible Retrieval und S3-Glacier Deep Archive müssen Sie eine Wiederherstellungsanforderung einleiten und warten, bis eine temporäre Kopie des Objekts verfügbar ist. Wenn eine temporäre Kopie des wiederhergestellten Objekts erstellt wird, bleibt die Speicherklasse des Objekts unverändert. (Eine HeadObject- oder GetObject-API-Operationsanforderung gibt S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive als Speicherklasse zurück.)
- Für Objekte in den Zugriffsebenen S3 Intelligent-Tiering Archive und S3 Intelligent-Tiering Deep Archive müssen Sie eine Wiederherstellungsanforderung einleiten und warten, bis das Objekt in die Stufe für häufigen Zugriff verschoben wird.

Weitere Informationen darüber, wie sich alle Amazon S3-Speicherklassen vergleichen, finden Sie unter Verwenden von Amazon-S3-Speicherklassen. Weitere Informationen zu S3 Intelligent-Tiering finden Sie unter the section called “So funktioniert S3 Intelligent-Tiering”.

Wiederherstellen von Objekten aus S3 Glacier

Wenn Sie S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive verwenden, stellt Amazon S3 eine temporäre Kopie des Objekts nur für die angegebene Dauer wieder her. Anschließend wird die Kopie des wiederhergestellten Objekts gelöscht. Sie können den Ablaufzeitraum einer wiederhergestellten Kopie durch die erneute Ausgabe einer Wiederherstellungsanforderung ändern. In diesem Fall aktualisiert Amazon S3 den Ablaufzeitraum relativ zum aktuellen Zeitraum.

> **Note**
>
> Bei einer Wiederherstellung eines archivierten Objekts von S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive zahlen Sie sowohl für das archivierte Objekt als auch für die temporär wiederhergestellte Kopie. Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

Wiederherstellen von Objekten aus S3 Intelligent-Tiering

Wenn Sie ein Objekt aus der S3 Intelligent-Tiering Archive Access Tier oder S3 Intelligent-Tiering Deep Archive Access Tier wiederherstellen, wird das Objekt zurück in die S3 Intelligent-Tiering Frequent Access Tier verschoben. Wenn dnach 30 aufeinanderfolgenden Tagen nicht auf das Objekt zugegriffen wird, wird es automatisch in Infrequent Access Tier verschoben. Das Objekt wechselt nach mindestens 90 aufeinanderfolgenden Tagen ohne Zugriff automatisch in die S3 Intelligent-Tiering Archive Access Tier. Wenn mindestens 180 aufeinanderfolgende Tage nicht auf ein Objekt zugegriffen wird, wird das Objekt in Deep Archive Access Tier verschoben.

> **Note**
>
> Im Gegensatz zu den Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive akzeptieren Wiederherstellungsanforderungen für S3-Intelligent-Tiering-Objekte den `Days`-Wert nicht.

Verwenden von S3-Batch-Operationen mit Wiederherstellungsanforderungen

Zur Wiederherstellung von mehr als einem Amazon S3-Objekt in einer einzigen Anforderung können Sie S3-Batch-Vorgänge nutzen. Sie stellen S3 Batch Operations eine Liste von Objekten zur Verfügung, für die Vorgänge ausgeführt werden sollen. S3-Batchoperationen rufen die entsprechende API-Operation auf, um die angegebene Operation auszuführen. Ein einzelner Batch-Vorgangsauftrag kann die angegebene Operation für Milliarden von Objekten ausführen, die Exabytes von Daten enthalten.

Wiederherstellungszeit

Amazon S3 berechnet die Ablaufzeit der wiederhergestellten Objektkopie durch Addition der Anzahl der in der Wiederherstellungsanforderung angegebenen Tage zum Zeitpunkt, wenn die angeforderte Wiederherstellung abgeschlossen wird. Anschließend rundet Amazon S3 die so berechnete Zeit auf den nächsten Tag um Mitternacht (UTC) auf. Beispiel: Angenommen, ein Objekt wurde am 15. Oktober 2012 um 10:30 Uhr UTC erstellt und als Wiederherstellungszeitraum wurden drei Tage angegeben. In diesem Fall läuft die wiederhergestellte Kopie am 19. Oktober 2012 um 00:00 Uhr UTC ab, zu welchem Zeitpunkt Amazon S3 die Objektkopie löscht.

Die Zeit, die für den Abschluss eines Wiederherstellungsauftrags benötigt wird, hängt davon ab, welche Archivspeicherklasse oder Speicherebene Sie verwenden und welche Abrufoption Sie angeben: Beschleunigt (nur für S3 Glacier Flexible Retrieval und S3 Intelligent Tiering Archive Access verfügbar), Standard, oder Bulk. Weitere Informationen finden Sie unter Archiv-Abrufoptionen.

Sie können über Amazon-S3-Ereignisbenachrichtigungen über den Abschluss der Wiederherstellung benachrichtigt werden. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Themen

Archiv-Abrufoptionen

Nachfolgend finden Sie die verfügbaren Abrufoptionen bei der Wiederherstellung eines archivierten Objekts in Amazon S3:

- Beschleunigt: Greifen Sie schnell auf Ihre Daten zu, die in der Speicherklasse „S3 Glacier Flexible Retrieval“ oder der S3 Intelligent-Tiering-Zugriffsebene „ Archive“ gespeichert sind. Sie können diese Option verwenden, wenn gelegentliche dringende Anforderungen für eine Teilmenge von Archiven erforderlich sind. Daten, die unter Verwendung von beschleunigten Abrufen abgerufen werden, stehen normalerweise innerhalb von 1 bis 5 Minuten zur Verfügung, außer es handelt sich um die größten archivierten Objekte (250 MB+).

  Die bereitgestellte Kapazität hilft sicherzustellen, dass für Expedited-Abrufe aus S3 Glacier Flexible Retrieval Abrufkapazität verfügbar ist, wenn Sie sie benötigen. Weitere Informationen finden Sie unter Bereitgestellte Kapazität.
- Standard: Greifen Sie auf Ihre archivierten Objekte innerhalb einiger Stunden zu. Standard ist die Standardoption für Abrufanforderungen, in denen keine Abrufoption angegeben ist. Die Standardabrufe enden in der Regel innerhalb von 3 bis 5 Stunden für Objekte, die in der S3 Glacier Flexible Retrieval Speicherklasse oder der S3 Intelligent Tiering Archive Access Stufe gespeichert sind. Diese Abrufe enden in der Regel innerhalb von 12 Stunden für Objekte, die in der Speicherklasse S3 Glacier Deep Archive oder S3 Intelligent-Tiering Deep Archive Access Tier gespeichert sind. Standardabrufe sind für Objekte, die in S3 Intelligent-Tiering gespeichert sind, kostenlos.

> **Note**
>
> Die Standardabrufe, die mit der Wiederherstellungsoperation von S3 Batch Operations gestartet werden, starten in der Regel innerhalb von wenigen Minuten und enden innerhalb von 3 bis 5 Stunden für Objekte, die in der S3 Glacier Flexible Retrieval Speicherklasse oder der S3 Intelligent Tiering Archive Access Stufe gespeichert sind. Die Standardabrufe, die mit Batch Operations für Objekte in der Speicherklasse S3 Glacier Deep Archive oder S3 Intelligent-Tiering Deep Archive Access gestartet wurden, beginnen in der Regel innerhalb von 9 Stunden und enden innerhalb von 12 Stunden.

- Bulk: Greifen Sie über die kostengünstigste Abrufoption von Amazon S3 Glacier auf Ihre Daten zu. Mit Massenabrufen können Sie Daten in großen Mengen (sogar im Petabyte-Bereich) kostengünstig abrufen. Die Massenabrufe werden in der Regel innerhalb von 5 bis 12 Stunden für Objekte beendet, die in der Speicherklasse S3 Glacier Flexible Retrieval oder S3 Intelligent Tiering Archive Access Tier gespeichert sind. Diese Abrufe enden in der Regel innerhalb von 48 Stunden für Objekte, die in der Speicherklasse S3 Glacier Deep Archive oder S3 Intelligent-Tiering Deep Archive Access Tier gespeichert sind. Massenabrufe sind für Objekte, die in S3 Glacier Flexible Retrieval und S3 Intelligent-Tiering gespeichert sind, kostenlos.

Die folgende Tabelle fasst die Archivabrufoptionen zusammen. Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

Wenn Sie einen `Expedited`-, `Standard`- oder `Bulk`-Abruf durchführen möchten, legen Sie das `Tier`-Anforderungselement in der REST-API-Operationsanforderung RestoreObject auf die gewünschte Option fest bzw. auf das Äquivalent in der AWS Command Line Interface (AWS CLI) oder den AWS-SDKs. Wenn Sie bereitgestellte Durchsatzkapazität gekauft haben, werden alle Expedited-Abrufe automatisch über Ihre bereitgestellte Kapazität erledigt.

Bereitgestellte Kapazität

Die bereitgestellte Kapazität hilft sicherzustellen, dass für Ihre Expedited-Abrufe aus S3 Glacier Flexible Retrieval Abrufkapazität verfügbar ist, wenn Sie sie benötigen. Jede Kapazitätseinheit stellt sicher, dass alle fünf Minuten mindestens drei beschleunigte Abrufe ausgeführt werden können, und bietet bis zu 150 MB/s Abrufdurchsatz.

Denken Sie über den Kauf bereitgestellter Abrufkapazität nach, wenn Ihr Workload einen sehr zuverlässigen und vorhersehbaren Zugriff auf eine Untermenge Ihrer Daten innerhalb von Minuten

erforderlich macht. Ohne bereitgestellte Kapazität werden beschleunigte Abrufe in Zeiten hoher Nachfrage möglicherweise nicht akzeptiert. Wenn Sie unbedingt Zugriff auf beschleunigte Abrufe benötigen, sollten Sie eine bereitgestellte Abrufkapazität kaufen.

Bereitgestellte Kapazitätseinheiten werden einem AWS-Konto zugewiesen. Daher sollte der Antragsteller des beschleunigten Datenabrufs die bereitgestellte Kapazitätseinheit erwerben, nicht der Bucket-Eigentümer.

Sie können bereitgestellte Kapazität über die Amazon-S3-Konsole, die Amazon-S3-Glacier-Konsole, die REST-API-Operation Purchase Provisioned Capacity, die AWS SDKs oder die AWS CLI kaufen. Weitere Informationen zu den Preisen für die bereitgestellte Kapazität finden Sie in der Amazon S3-Preisliste.

Geschwindigkeiten für Wiederherstellungsanforderungen für S3 Glacier

Wenn Sie Wiederherstellungsanforderungen für Objekte initiieren, die in den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeichert sind, wird ein Kontingent für Wiederherstellungsanforderungen für Ihr AWS-Konto angewendet. S3 Glacier unterstützt Wiederherstellungsanforderungen mit einer Geschwindigkeit von 1 000 Transaktionen pro Sekunde. Wenn diese Rate überschritten wird, werden andernfalls gültige Anfragen gedrosselt oder zurückgewiesen und Amazon S3 gibt einen `ThrottlingException`-Fehler zurück.

Optional können Sie auch S3-Massenoperationen verwenden, um eine große Anzahl von Objekten, die in S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeichert sind, mit einer einzigen Anforderung abzurufenu. Weitere Informationen finden Sie unter Grundlagen von S3-BatchVorgänge.

Wiederherstellen eines archivierten Objekts

Amazon-S3-Objekte in den folgenden Speicherklassen oder -stufen werden archiviert. Es kann nicht in Echtzeit darauf zugegriffen werden:

- Die Speicherklasse S3 Glacier Flexible Retrieval
- Die Speicherklasse S3 Glacier Deep Archive
- Die Zugriffsebene S3 Intelligent-Tiering Archive
- Die S3-Intelligent-Tiering-Deep-Archive-Zugriffsebene

Ein Zugriff auf in der Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeicherte Amazon-S3-Objekte ist nicht unmittelbar möglich. Um auf ein Objekt in diesen Speicherklassen zuzugreifen, müssen Sie eine temporäre Kopie des Objekts für eine angegebene

Dauer (Anzahl von Tagen) in seinem S3-Bucket wiederherstellen. Wenn Sie eine permanente Kopie des Objekts benötigen, stellen Sie das Objekt wieder her und legen Sie dann eine Kopie davon in Ihrem Amazon S3-Bucket an. Das Kopieren wiederhergestellter Objekte wird in der Amazon-S3-Konsole nicht unterstützt. Verwenden Sie für diese Art von Kopiervorgang die AWS Command Line Interface (AWS CLI), die AWS-SDKs oder die REST-API. Wenn Sie keine Kopie erstellen und die Speicherklasse ändern, wird das Objekt weiterhin in den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeichert. Informationen zur Verwendung dieser Speicherklassen finden Sie unter Speicherklassen für die Archivierung von Objekten.

Für Zugriff auf Objekte in den S3 Intelligent-Tiering Archive Access und Deep Archive Access müssen Sie eine Wiederherstellungsanforderung initiieren und warten, bis das Objekt in die Stufe für häufigen Zugriff verschoben wird, um darauf zuzugreifen. Wenn Sie ein Objekt aus den Stufen Archive Access oder Deep Archive Access wiederherstellen, wird das Objekt zurück in die Stufe für häufige Zugriffe übergehen. Informationen zur Verwendung dieser Speicherklassen finden Sie unter Speicherklasse zur automatischen Optimierung von Daten mit sich ändernden oder unbekannten Zugriffsmustern.

Allgemeine Informationen über archivierte Objekte finden Sie unter Arbeiten mit archivierten Objekten.

> **Note**
>
> Bei einer Wiederherstellung eines archivierten Objekts aus S3 Glacier zahlen Sie sowohl für das archivierte Objekt als auch für die temporär wiederhergestellte Kopie. Wenn Sie ein Objekt aus S3 Intelligent-Tiering wiederherstellen, fallen keine Abrufgebühren für Standard- oder Bulk-Abrufe an. Nachfolgende Wiederherstellungsanforderungen, die für archivierte Objekte aufgerufen werden, die bereits wiederhergestellt werden, werden jedoch als GET-Anforderung in Rechnung gestellt. Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

Wiederherstellen eines archivierten Objekts

Sie können ein archiviertes Objekt mithilfe der Amazon-S3-Konsole, der REST-API, den AWS SDKs, der AWS Command Line Interface (AWS CLI) oder S3 Batch Operations wiederherstellen.

Verwenden der S3-Konsole

Wiederherstellung von Objekten mithilfe der Amazon-S3-Konsole

Gehen Sie wie folgt vor, um ein Objekt wiederherzustellen, das in den Speicherklassen „S3 Glacier Flexible Retrieval“ oder „S3 Glacier Deep Archive“ bzw. in den Speicherstufen „S3 Intelligent-Tiering Archive Access“ oder „Deep Archive Access“ archiviert wurde.

So stellen Sie ein archiviertes Objekt wieder her

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der die Objekte enthält, die Sie wiederherstellen möchten.
4. Wählen Sie in der Liste Object (Objekt) das/die Objekt(e) aus, das/die Sie wiederherstellen möchten. Wählen Sie anschließend Actions (Aktionen) und dann Initiate restore (Wiederherstellung initiieren) aus.
5. Wenn Sie von S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive wiederherstellen, geben Sie im Dialogfeld Anzahl der Tage, die die wiederhergestellte Kopie verfügbar ist die Anzahl der Tage ein, die die archivierten Daten verfügbar sein sollen.
6. Führen Sie bei Abrufoptionen einen der folgenden Schritte aus:

   - Wählen Sie Massenabruf oder Standardabruf und dann Wiederherstellen starten aus.
   - Wählen Sie Expedited retrieval (Expedited-Abruf) (nur für S3 Glacier Flexible Retrieval oder S3 Intelligent Tiering Archive Access verfügbar). Wenn Sie ein Objekt in S3 Glacier Flexible Retrieval wiederherstellen, können Sie auswählen, ob Sie bereitgestellte Kapazität für Ihren Expressabruf erwerben möchten. Wenn Sie bereitgestellte Kapazität erwerben möchten, fahren Sie mit dem nächsten Schritt fort. Wenn Sie dies nicht wünschen, wählen Sie Wiederherstellen starten aus.
7. (Optional) Wenn Sie ein Objekt in S3 Glacier Flexible Retrieval wiederherstellen und Beschleunigter Abruf auswählen, können Sie sich entscheiden, ob Sie bereitgestellte Kapazität erwerben möchten. Die bereitgestellte Kapazität steht nur für Objekte in S3 Glacier Flexible Retrieval zur Verfügung. Wenn Sie bereits über bereitgestellte Kapazität verfügen, wählen Sie Wiederherstellen starten aus, um einen bereitgestellten Abruf zu starten.

   Wenn Sie bereitgestellte Durchsatzkapazität gekauft haben, werden alle Ihre Expedited-Abrufe automatisch über Ihre bereitgestellte Kapazität erledigt. Weitere Informationen finden Sie unter Bereitgestellte Kapazität.

- Wenn Sie nicht über bereitgestellte Kapazität verfügen und auch keine kaufen möchten, wählen Sie Wiederherstellen starten aus.
- Wenn Sie keine bereitgestellte Kapazität haben, aber bereitgestellte Kapazitätseinheiten (PCUs) kaufen möchten, wählen Sie PCUs kaufen aus. Wählen Sie im Dialogfeld PCUs kaufen aus, wie viele PCUs Sie kaufen möchten, bestätigen Sie Ihren Kauf und wählen Sie dann PCUs kaufen aus. Wenn Sie die Meldung Kauf erfolgreich erhalten, wählen Sie Wiederherstellen starten aus, um den bereitgestellten Abruf zu starten.

Verwendung von AWS CLI

Objekte aus S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive wiederherstellen

Im folgenden Beispiel wird mit dem `restore-object`-Befehl das Objekt *dir1/example.obj* 25 Tage lang in Bucket *DOC-EXAMPLE-BUCKET* wiederhergestellt.

```
aws s3api restore-object --bucket DOC-EXAMPLE-BUCKET --key dir1/example.obj --restore-request '{"Days":25,"GlacierJobParameters":{"Tier":"Standard"}}'
```

Wenn die im Beispiel verwendete JSON-Syntax zu einem Fehler auf einem Windows-Client führt, ersetzen Sie die Wiederherstellungsanforderung durch die folgende Syntax:

```
--restore-request Days=25,GlacierJobParameters={"Tier"="Standard"}
```

Objekte aus S3 Intelligent-Tiering Archive Access und Deep Archive Access wiederherstellen

Im folgenden Beispiel wird mit dem `restore-object`-Befehl das Objekt *dir1/example.obj* im -Bucket *DOC-EXAMPLE-BUCKET* auf der Stufe Frequent Access wiederhergestellt.

```
aws s3api restore-object --bucket DOC-EXAMPLE-BUCKET --key dir1/example.obj --restore-request '{}'
```

Den Wiederherstellungsstatus überwachen

Sie können den folgenden `head-object`-Befehl verwenden, um den Status Ihrer `restore-object`-Anforderung zu überwachen:

```
aws s3api head-object --bucket DOC-EXAMPLE-BUCKET --key dir1/example.obj
```

Weitere Informationen finden Sie unter restore-object in der Referenz zum AWS CLI-Befehl.

Verwenden der REST-API

Amazon S3 stellt einen API-Vorgang für Sie bereit, um die Wiederherstellung eines Archivs zu starten. Weitere Informationen finden Sie unter RestoreObject in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS-SDKs

Beispiele für die Wiederherstellung archivierter Objekte in S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive mit AWS SDKs finden Sie unter Wiederherstellen einer archivierten Kopie eines Objekts in einem Amazon S3-Bucket mithilfe eines - AWS SDK.

Verwendung von S3 Batch Operations

Zur Wiederherstellung von mehr als einem archivierten Objekt in einer einzigen Anforderung können Sie S3 Batch Operations nutzen. Sie stellen S3 Batch Operations eine Liste von Objekten zur Verfügung, für die Vorgänge ausgeführt werden sollen. S3-Batchoperationen rufen die entsprechende API-Operation auf, um die angegebene Operation auszuführen. Ein einzelner Batch-Vorgangsauftrag kann die angegebene Operation für Milliarden von Objekten ausführen, die Exabytes von Daten enthalten.

Um einen Batch-Operations-Auftrag zu erstellen, benötigen Sie ein Manifest, das nur die Objekte enthält, die Sie wiederherstellen möchten. Sie können mithilfe von S3 Inventory ein Manifest erstellen oder eine CSV-Datei mit den erforderlichen Informationen bereitstellen. Weitere Informationen finden Sie unter the section called “Angeben eines Manifests”.

Bevor Sie S3-Batch-Operations-Aufträge erstellen und ausführen, müssen Sie Amazon S3 die Erlaubnis erteilen, S3-Batch-Operations in Ihrem Namen durchzuführen. Die erforderlichen Berechtigungen finden Sie unter the section called “Gewähren von Berechtigungen”.

> **Note**
>
> Batch-Operations-Aufträge können entweder mit Objekte der Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive oder mit Objekten der Speicherstufen S3 Intelligent-Tiering Archive Access und Deep Archive Access arbeiten. Batch Operations kann im selben Auftrag nicht für beide Typen von archivierten Objekten ausgeführt werden. Um Objekte beider Typen wiederherzustellen, müssen Sie separate Batchoperations-Aufgaben erstellen.

Weitere Informationen zur Verwendung von Batch Operations zum Wiederherstellen archivierter Objekte finden Sie unter the section called “Wiederherstellen von Objekten”.

So erstellen Sie einen S3-Auftrag „Initiate Restore Object Batch Operations“

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie im linken Navigationsbereich auf Batchvorgänge.
3. Wählen Sie Create job (Auftrag erstellen) aus.
4. Wählen Sie für AWS-Region die Region aus, in der Sie Ihren Auftrag erstellen möchten.
5. Wählen Sie unter Manifestformat das zu verwendende Manifest aus.
   - Wenn Sie S3-Bestandsbericht auswählen, geben Sie den Pfad zum `manifest.json`-Objekt ein, das Amazon S3 als Teil des Bestandsberichts im CSV-Format generiert hat. Wenn Sie nicht das aktuelle Manifest verwenden möchten, können Sie optional die Versions-ID des `manifest.json`-Objekts angeben.
   - Wenn Sie CSV auswählen, geben Sie den Pfad zu einem CSV-formatierten Manifestobjekt ein. Das Manifestobjekt muss das in der Konsole beschriebene Format befolgen. Wenn Sie nicht die aktuelle Version verwenden möchten, können Sie optional die Versions-ID des Manifestobjekts angeben.
6. Wählen Sie Next (Weiter).
7. Wählen Sie im Abschnitt Operation die Option Wiederherstellen aus.
8. Wählen Sie im Abschnitt Wiederherstellen bei Quelle wiederherstellen entweder Glacier Flexible Retrieval oder Glacier Deep Archive bzw. die Stufe Intelligent-Tiering Archive Access oder Deep Archive Access aus.

   Wenn Sie Glacier Flexible Retrieval oder Glacier Deep Archive auswählen, geben Sie eine Zahl für die Anzahl der Tage, an denen die wiederhergestellte Kopie verfügbar ist ein.

   Wählen Sie bei Abrufstufe die zu verwendende Stufe aus.
9. Wählen Sie Next (Weiter).
10. Füllen Sie auf der Seite Zusätzliche Optionen konfigurieren die folgenden Abschnitte aus:

- Geben Sie im Abschnitt Zusätzliche Optionen eine Beschreibung des Auftrags und eine Prioritätsnummer für den Auftrag an. Höhere Nummern bedeuten eine höhere Priorität. Weitere Informationen finden Sie unter the section called “Zuweisen der Auftragspriorität”.
- Wählen Sie im Abschnitt Abschlussbericht aus, ob Batch Operations einen Abschlussbericht erstellen soll. Weitere Informationen zu den Fertigstellungsberichten finden Sie unter the section called “Abschlussberichte”.
- Im Abschnitt Berechtigungen müssen Sie Amazon S3 die Erlaubnis erteilen, Batch Operations in Ihrem Namen durchzuführen. Die erforderlichen Berechtigungen finden Sie unter the section called “Gewähren von Berechtigungen”.
- (Optional) fügen Sie im Abschnitt Auftrags-Tags Tags in Schlüssel-Wert-Paaren hinzu. Weitere Informationen finden Sie unter the section called “Verwenden von Markierungen”.

Wählen Sie Weiter aus, sobald Sie fertig sind.

11. Überprüfen Sie die Einstellungen auf der Seite Review. Wenn Sie Änderungen vornehmen müssen, wählen Sie Previous. Wählen Sie andernfalls Auftrag erstellen.

Weitere Informationen über Batch Operations finden Sie unter Objekte mit Batch Operations wiederherstellen und Erstellen eines S3-Batch-Vorgangsauftrags.

Überprüfung des Wiederherstellungsstatus und des Ablaufdatums

Sie können den Status einer Wiederherstellungsanfrage oder das Ablaufdatum mithilfe der Amazon-S3-Konsole, der AWS CLI oder der REST-API überprüfen.

> **Note**
>
> Wiederhergestellte Objekte von S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive werden nur für die von Ihnen angegebene Anzahl von Tagen gespeichert. Mit den folgenden Verfahren wird das Ablaufdatum für diese Kopien zurückgegeben. Objekte, die aus den Speicherstufen „S3 Intelligent-Tiering Archive Access“ und „Deep Archive Access“ wiederhergestellt wurden, haben kein Ablaufdatum und werden stattdessen wieder in die Stufe „Frequent Zugriff“ verschoben.

Verwenden der S3-Konsole

So überprüfen Sie den Wiederherstellungsstatus und das Ablaufdatum eines Objekts in der Amazon-S3-Konsole

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der die Objekte enthält, die Sie wiederherstellen.
4. Wählen Sie in der Liste Objekte das Objekt aus, das Sie wiederherstellen. Die Seite mit den Objektdetails wird angezeigt.

    - Wenn die Wiederherstellung noch nicht abgeschlossen ist, wird oben auf der Seite der Abschnitt Wiederherstellung in Bearbeitung angezeigt.
    - Wenn die Wiederherstellung abgeschlossen ist, wird oben auf der Seite der Abschnitt Wiederherstellung abgeschlossen angezeigt. Wenn Sie aus S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive wiederherstellen, werden in diesem Abschnitt auch das Ablaufdatum der Wiederherstellung angezeigt. Amazon S3 entfernt die wiederhergestellte Kopie aus dem archivierten Objekt an diesem Datum.

Verwendung von AWS CLI

Den Wiederherstellungsstatus und das Ablaufdatum eines Objekts mit AWS CLI überprüfen

Im folgenden Beispiel wird der `head-object`-Befehl verwendet, um Metadaten für das Objekt *`dir1/example.obj`* im Bucket *`DOC-EXAMPLE-BUCKET`* anzuzeigen. Wenn Sie diesen Befehl für ein wiederhergestelltes Objekt ausführen, gibt Amazon S3 zurück, ob die Wiederherstellung noch läuft und (falls zutreffend) das Ablaufdatum.

```
aws s3api head-object --bucket DOC-EXAMPLE-BUCKET --key dir1/example.obj
```

Erwartete Ausgabe (Wiederherstellung läuft):

```
{
    "Restore": "ongoing-request=\"true\"",
    "LastModified": "2020-06-16T21:55:22+00:00",
    "ContentLength": 405,
    "ETag": "\"b662d79adeb7c8d787ea7eafb9ef6207\"",
```

```
    "VersionId": "wbYaE2vtOV0iIBXrOqGAJt3fP1cHB8Wi",
    "ContentType": "binary/octet-stream",
    "ServerSideEncryption": "AES256",
    "Metadata": {},
    "StorageClass": "GLACIER"
}
```

Erwartete Ausgabe (Wiederherstellung abgeschlossen):

```
{
    "Restore": "ongoing-request=\"false\", expiry-date=\"Wed, 12 Aug 2020 00:00:00 GMT
\"",
    "LastModified": "2020-06-16T21:55:22+00:00",
    "ContentLength": 405,
    "ETag": "\"b662d79adeb7c8d787ea7eafb9ef6207\"",
    "VersionId": "wbYaE2vtOV0iIBXrOqGAJt3fP1cHB8Wi",
    "ContentType": "binary/octet-stream",
    "ServerSideEncryption": "AES256",
    "Metadata": {},
    "StorageClass": "GLACIER"
}
```

Weitere Informationen zu `head-object` finden Sie unter head-object in der AWS CLI-Referenz.

Verwenden der REST-API

Amazon S3 bietet eine API-Operation, mit der Sie Objektmetadaten abrufen können. Informationen zum Überprüfen des Wiederherstellungsstatus und des Ablaufdatums eines archivierten Objekts mithilfe der REST-API finden Sie unter HeadObject in der Referenz zur API für Amazon Simple Storage Service.

Upgraden der Geschwindigkeit einer Wiederherstellung in Bearbeitung

Sie können die Geschwindigkeit einer Wiederherstellung in Bearbeitung upgraden.

So upgraden Sie eine Wiederherstellung in Bearbeitung auf eine schnellere Stufe:

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der die Objekte enthält, die Sie wiederherstellen möchten.

4. Wählen Sie in der Liste Objekte das Objekt aus, das Sie wiederherstellen. Die Seite mit den Objektdetails wird angezeigt. Wählen Sie auf der Detailseite des Objekts Upgrade-Abrufstufe. Informationen zum Überprüfen des Wiederherstellungsstatus eines Objekts finden Sie unter Überprüfung des Wiederherstellungsstatus und des Ablaufdatums.
5. Wählen Sie die Stufe aus, auf die Sie upgraden möchten, und dann Wiederherstellen starten aus.

Verwenden der S3-Objektsperre

Mit S3 Object Lock können Sie für einen bestimmten Zeitraum oder auf unbestimmte Zeit verhindern, dass Amazon-S3-Objekte gelöscht oder überschrieben werden. Object Lock verwendet ein write-once-read-many (WORM)-Modell zum Speichern von Objekten. Sie können die Objektsperre verwenden, um regulatorische Anforderungen zu erfüllen, die WORM-Speicher erfordern, oder um eine weitere Schutzebene vor Objektänderungen oder -löschungen hinzuzufügen.

> **Note**
>
> S3 Object Lock wurde von Cohasset Associates in Bezug auf die Verwendung in Umgebungen bewertet, die den Bestimmungen von SEC 17a-4, CFTC und FINRA unterliegen. Weitere Informationen zu Object Lock im Zusammenhang mit diesen Bestimmungen finden Sie unter Cohasset AssociatesCompliance-Bewertung.

Die Objektsperre stellt zwei Optionen für die Verwaltung der Aufbewahrung von Objekten bereit: Aufbewahrungszeiträume und rechtliche Aufbewahrungsfristen. Für eine Objektversion kann es sowohl einen Aufbewahrungszeitraum als auch eine rechtliche Aufbewahrungsfrist geben oder beides geben.

- Aufbewahrungszeitraum – Ein Aufbewahrungszeitraum gibt einen festen Zeitraum an, für den ein Objekt gesperrt bleibt. Sie können einen eindeutigen Aufbewahrungszeitraum für einzelne Objekte festlegen. Darüber hinaus können Sie einen Standardaufbewahrungszeitraum für einen S3-Bucket festlegen. Sie können auch die minimal und maximal zulässigen Aufbewahrungszeiträume mit dem `s3:object-lock-remaining-retention-days` Bedingungsschlüssel in der Bucket-Richtlinie einschränken. Auf diese Weise können Sie einen Aufbewahrungszeitraum festlegen und Aufbewahrungszeiträume einschränken, die kürzer oder länger als dieser Bereich sein können.
- Gesetzliche Aufbewahrungsfrist – Eine gesetzliche Aufbewahrungsfrist bietet denselben Schutz wie ein Aufbewahrungszeitraum, hat jedoch kein Ablaufdatum. Stattdessen bleibt eine rechtliche

Aufbewahrungsfrist gültig, bis Sie diese ausdrücklich entfernen. Rechtliche Aufbewahrungsfristen sind unabhängig von Aufbewahrungszeiträumen und werden auf einzelne Objektversionen angewendet.

Object Lock funktioniert nur in Buckets, für die S3 Versioning aktiviert ist. Wenn Sie eine Objektversion sperren, speichert Amazon S3 die Sperrinformationen in den Metadaten für diese Objektversion. Die Platzierung eines Aufbewahrungszeitraums oder einer rechtlichen Aufbewahrungsfrist für ein Objekt schützt ausschließlich die in der Anforderung angegebene Version. Aufbewahrungszeiträume und rechtliche Aufbewahrungsfristen verhindern nicht, dass neue Versionen des Objekts erstellt oder Löschmarkierungen zusätzlich zum Objekt hinzugefügt werden. Weitere Informationen über S3 Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

Wenn Sie ein Objekt in einem Bucket platzieren, der bereits ein vorhandenes, geschütztes Objekt mit demselben Objektschlüsselnamen enthält, erstellt Amazon S3 eine neue Version dieses Objekts. Die vorhandene, geschützte Version des Objekts bleibt entsprechend seiner Aufbewahrungskonfiguration gesperrt.

So funktioniert die S3-Objektsperre

Themen

Aufbewahrungszeiträume

Ein Aufbewahrungszeitraum schützt eine Objektversion für einen festen Zeitraum. Wenn Sie für eine Objektversion einen Aufbewahrungszeitraum festlegen, speichert Amazon S3 einen Zeitstempel in den Metadaten der Objektversion, um anzugeben, wann der Aufbewahrungszeitraum abläuft. Nach Ablauf des Aufbewahrungszeitraums kann die Objektversion überschrieben oder gelöscht werden.

Sie können eine Aufbewahrungsfrist explizit für eine einzelne Objektversion oder für die Eigenschaften eines Buckets festlegen, so dass sie automatisch für alle Objekte in dem Bucket gilt. Wenn Sie einen Aufbewahrungszeitraum explizit auf eine Objektversion anwenden, geben Sie ein

Retain Until Date (Bis-Aufbewahrungsdatum) für die Objektversion an. Amazon S3 speichert dieses Datum in den Metadaten der Objektversion.

Sie können auch eine Aufbewahrungsfrist in den Eigenschaften eines Buckets festlegen. Wenn Sie eine Aufbewahrungsfrist für einen Bucket festlegen, geben Sie eine Dauer in Tagen oder Jahren an, für die jede im Bucket platzierte Objektversion geschützt werden soll. Wenn Sie ein Objekt in den Bucket legen, berechnet Amazon S3 ein Bis-Aufbewahrungsdatum für die Objektversion, indem es die angegebene Dauer dem Zeitstempel der Erstellung der Objektversion hinzufügt. Die Objektversion wird anschließend so geschützt, als ob Sie für die Objektversion ausdrücklich eine individuelle Sperre mit diesem Aufbewahrungszeitraum angegeben hätten.

> **Note**
>
> Wenn Sie PUT für eine Objektversion mit einem expliziten individuellen Aufbewahrungsmodus und Zeitraum in einem Bucket verwenden, haben die individuellen Object-Lock-Einstellungen der Objektversion Vorrang vor allen Aufbewahrungseinstellungen für Bucket-Eigenschaften.

Wie alle anderen Einstellungen für die Objektsperre gelten Aufbewahrungszeiträume für einzelne Objektversionen. Für verschiedene Versionen desselben Objekts können verschiedene Aufbewahrungsmodi und -zeiträume gelten.

z. B.: Angenommen, Sie haben ein Objekt, von dessen 30-tägigem Aufbewahrungszeitraum 15 Tage abgelaufen sind, und Sie führen die Aktion PUT für ein Objekt in Amazon S3 mit demselben Namen und einem 60-tägigen Aufbewahrungszeitraum durch. In diesem Fall ist die PUT-Anforderung erfolgreich und Amazon S3 erstellt eine neue Version des Objekts mit einem 60-tägigen Aufbewahrungszeitraum. Die ältere Version behält den ursprünglichen Aufbewahrungszeitraum und kann in 15 Tagen gelöscht werden.

Sie können einen Aufbewahrungszeitraum verlängern, nachdem Sie eine Aufbewahrungseinstellung auf eine Objektversion angewendet haben. Senden Sie dazu eine neue Object-Lock-Anforderung für die Objektversion mit einem Aufbewahren bis-Datum, das später liegt als das derzeit für die Objektversion konfigurierte Datum. Amazon S3 ersetzt die bestehende Aufbewahrungsfrist durch den neuen, längeren Zeitraum. Alle Benutzer, die die Berechtigung zur Festlegung von Aufbewahrungszeiträumen für Objekte besitzen, können Aufbewahrungszeiträume für Objektversionen verlängern. Um Aufbewahrungszeitraum festzulegen, müssen Sie die `s3:PutObjectRetention`-Berechtigung besitzen.

Wenn Sie einen Aufbewahrungszeitraum für ein Objekt oder einen S3-Bucket festlegen, müssen Sie einen von zwei Aufbewahrungsmodi auswählen: Compliance oder Governance.

Aufbewahrungsmodi

S3 Object Lock bietet zwei Aufbewahrungsmodi mit unterschiedlichen Schutzebenen für Ihre Objekte:

- Compliance-Modus
- Governance-Modus

Im Compliance-Modus kann eine geschützte Objektversion von keinem Benutzer überschrieben oder gelöscht werden. Dies schließt den Stammbenutzer in Ihrem AWS-Konto ein. Wenn ein Objekt im Compliance-Modus gesperrt wurde, können der Aufbewahrungsmodus nicht geändert und der Aufbewahrungszeitraum nicht verkürzt werden. Der Compliance-Modus stellt sicher, dass eine Objektversion während des Aufbewahrungszeitraums weder überschrieben noch gelöscht werden.

> **Note**
>
> Die einzige Möglichkeit, ein Objekt im Compliance-Modus vor Ablauf des Aufbewahrungsdatums zu löschen, besteht darin, das zugehörige zu löschen AWS-Konto.

Im Governance-Modus können Benutzer eine Objektversion nicht überschreiben oder löschen oder ihre Sperreinstellungen ändern, wenn sie keine speziellen Berechtigungen besitzen. Mit dem Governance-Modus schützen Sie Objekte dagegen, von den meisten Benutzern gelöscht zu werden, Sie können jedoch weiterhin einigen Benutzern die Berechtigung geben, die Aufbewahrungseinstellungen zu ändern oder die Objekte bei Bedarf zu löschen. Sie können den Governance-Modus auch verwenden, um die Einstellungen für Aufbewahrungszeiträume zu testen, bevor Sie einen Aufbewahrungszeitraum im Compliance-Modus erstellen.

Um Aufbewahrungseinstellungen im Governance-Modus zu überschreiben oder zu entfernen, müssen Sie die Berechtigung `s3:BypassGovernanceRetention` besitzen und in jede Anfrage, die ein Überschreiben des Governance-Modus erfordert, ausdrücklich `x-amz-bypass-governance-retention:true` als Anfrage-Header einfügen.

> **Note**
>
> Die Amazon-S3-Konsole enthält standardmäßig den `x-amz-bypass-governance-retention:true`-Header. Wenn Sie versuchen, Objekte zu löschen, die durch den Governance-Modus geschützt sind und über die `s3:BypassGovernanceRetention`-Berechtigung verfügen, verläuft der Vorgang erfolgreich.

Rechtliche Aufbewahrungsfristen

Mit Object Lock können Sie auch eine gesetzliche Aufbewahrungsfrist für eine Objektversion festlegen. Wie Aufbewahrungszeiträume verhindern auch rechtliche Aufbewahrungsfristen das Überschreiben oder Löschen von Objektversionen. Mit gesetzlichen Aufbewahrungsfristen sind jedoch keine Aufbewahrungszeiträume verknüpft. Sie sind gültig, bis sie entfernt werden. Alle Benutzer mit der Berechtigung `s3:PutObjectLegalHold` können rechtliche Aufbewahrungsfristen festlegen und entfernen.

Rechtliche Aufbewahrungsfristen sind von Aufbewahrungszeiträumen unabhängig. Die Festlegung einer rechtlichen Aufbewahrungsfrist für eine Objektversion hat keine Auswirkungen auf den Aufbewahrungsmodus oder -zeitraum für die betreffende Objektversion.

Angenommen, Sie setzen eine gesetzliche Aufbewahrungsfrist für eine Objektversion fest, während diese Objektversion gleichzeitig durch einen Aufbewahrungszeitraum geschützt ist. Wenn der Aufbewahrungszeitaum abgläuft, verliert das Objekt seinen WORM-Schutz nicht. Stattdessen wird das Objekt durch die gesetzliche Aufbewahrungsfrist weiter geschützt, bis ein autorisierter Benutzer diese ausdrücklich entfernt. Wenn Sie eine rechtliche Aufbewahrungsfrist entfernen, während für eine Objektversion ein Aufbewahrungszeitraum gilt, bleibt die Objektversion geschützt, bis der Aufbewahrungszreitraum abläuft.

Bewährte Methoden für die Verwendung der S3-Objektsperre

Erwägen Sie die Verwendung des Governance-Modus, wenn Sie Objekte während eines vordefinierten Aufbewahrungszeitraums vor dem Löschen durch die meisten Benutzer schützen möchten, aber gleichzeitig möchten, dass einige Benutzer mit besonderen Berechtigungen die Flexibilität haben, die Aufbewahrungseinstellungen zu ändern oder die Objekte zu löschen.

Erwägen Sie die Verwendung des Compliance-Modus, wenn Sie niemals möchten, dass ein Benutzer, einschließlich des Root-Benutzers in Ihrem AWS-Konto, die Objekte während eines

vordefinierten Aufbewahrungszeitraums löschen kann. Sie können diesen Modus verwenden, falls Sie konforme Daten speichern müssen.

Sie können die gesetzliche Aufbewahrungsfrist verwenden, wenn Sie nicht sicher sind, wie lange Ihre Objekte unveränderlich bleiben sollen. Dies könnte daran liegen, dass Sie ein bevorstehendes externes Audit Ihrer Daten haben und Objekte unveränderlich halten möchten, bis das Audit abgeschlossen ist. Alternativ können Sie ein laufendes Projekt haben, das einen Datensatz verwendet, der unveränderlich bleiben soll, bis das Projekt abgeschlossen ist.

Erforderliche Berechtigungen

ObjektsperrenVorgänge erfordern bestimmte Berechtigungen. Abhängig von dem genauen Vorgang, den Sie versuchen, benötigen Sie möglicherweise eine der folgenden Berechtigungen:

- `s3:BypassGovernanceRetention`
- `s3:GetBucketObjectLockConfiguration`
- `s3:GetObjectLegalHold`
- `s3:GetObjectRetention`
- `s3:PutBucketObjectLockConfiguration`
- `s3:PutObjectLegalHold`
- `s3:PutObjectRetention`

Eine vollständige Liste der Amazon S3-Berechtigungen mit Beschreibungen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Informationen zur Verwendung von Bedingungen mit Berechtigungen finden Sie unter Beispiele für Amazon-S3-Bedingungsschlüssel.

Überlegungen zu Object Lock

Amazon S3 Object Lock kann verhindern, dass Objekte für einen bestimmten Zeitraum oder auf unbestimmte Zeit gelöscht oder überschrieben werden.

Sie können die Amazon S3-Konsole, AWS Command Line Interface (AWS CLI), AWS SDKs oder die Amazon S3-REST-API verwenden, um Informationen zur Objektsperre anzuzeigen oder festzulegen. Weitere Informationen zu den Funktionen von S3 Object Lock finden Sie unter Verwenden der S3-Objektsperre.

> **Important**
>
> - Wenn Sie Object Lock für einen Bucket aktiviert haben, können Sie Object Lock nicht deaktivieren oder die Versionsverwaltung für diesen Bucket aussetzen.
> - S3-Buckets mit Object Lock können nicht als Ziel-Buckets für Server-Zugriffsprotokolle verwendet werden. Weitere Informationen finden Sie unter the section called “Protokollierungs-Serverzugriff”.

Themen

Berechtigungen zum Anzeigen von Sperrinformationen

Sie können den Object-Lock-Status einer Amazon-S3-Objektversion mithilfe der Vorgänge HeadObject und GetObject anzeigen. Beide Vorgänge geben den Aufbewahrungsmodus, die Aufbewahrungsfrist und den Status in Bezug auf eine rechtliche Aufbewahrungsfrist für die angegebene Objektversion an. Darüber hinaus können Sie den Objektsperrenstatus für mehrere Objekte in Ihrem S3-Bucket mit S3 Inventory anzeigen.

Um Aufbewahrungsmodus und Aufbewahrungszeitraum für eine Objektversion anzuzeigen, müssen Sie die Berechtigung `s3:GetObjectRetention` besitzen. Um den Status eines Objekts in Bezug auf rechtliche Aufbewahrungsfristen anzuzeigen, müssen Sie die Berechtigung `s3:GetObjectLegalHold` besitzen. Um die Standardaufbewahrungskonfiguration eines Buckets

anzuzeigen, benötigen Sie die `s3:GetBucketObjectLockConfiguration`-Berechtigung. Wenn Sie eine Object-Lock-Konfiguration für einen Bucket anfordern, für den S3 Object Lock nicht aktiviert ist, gibt Amazon S3 einen Fehler zurück.

Umgehen des Governance-Modus

Sie können für im Governance-Modus gesperrte Objektversionen, Vorgänge ausführen, als ob sie nicht geschützt wären, wenn Sie die `s3:BypassGovernanceRetention`-Berechtigung besitzen. Zu diesen Vorgängen gehören das Löschen einer Objektversion, das Verkürzen des Aufbewahrungszeitraums oder das Entfernen des Object-Lock-Aufbewahrungszeitraums durch die Platzierung einer neuen `PutObjectRetention`-Anforderung mit leeren Parametern.

Um den Governance-Modus umgehen zu können, müssen Sie in Ihrer Anforderung ausdrücklich angeben, dass Sie den Governance-Modus umgehen möchten. Fügen Sie dazu den `-x-amz-bypass-governance-retention:true`Header in Ihre `PutObjectRetention` API-Operationsanforderung ein oder verwenden Sie den entsprechenden Parameter für Anforderungen, die über die AWS CLI oder AWS SDKs gestellt werden. Die S3-Konsole übernimmt diesen Header automatisch für über die S3-Konsole gestellte Anforderungen, wenn Sie über die `s3:BypassGovernanceRetention`-Berechtigung verfügen, den Governance-Modus zu umgehen.

> **Note**
>
> Die Umgehung des Governance-Modus hat keine Auswirkungen auf den Status einer Objektversion in Bezug auf rechtliche Aufbewahrungsfristen. Wenn für eine Objektversion eine rechtliche Aufbewahrungsfrist aktiviert ist, bleibt diese in Kraft und verhindert, dass die Objektversion durch Anforderungen überschrieben oder gelöscht wird.

Verwenden von Object Lock mit der S3-Replikation

Sie können Object Lock mit S3-Replikation verwenden, um automatisches asynchrones Kopieren von gesperrten Objekten und deren Aufbewahrungs-Metadaten über S3-Buckets hinweg zu aktivieren. Das bedeutet, dass Amazon S3 für replizierte Objekte die Objektsperrenkonfiguration des Quell-Buckets übernimmt. Mit anderen Worten, wenn für den Quell-Bucket die Objektsperre aktiviert ist, muss für die Ziel-Buckets auch die Objektsperre aktiviert sein. Wenn ein Objekt direkt in den Ziel-Bucket (außer S3 Replication) hochgeladen wird, wird die Objektsperre für den Ziel-Bucket verwendet. Wenn Sie Replikation verwenden, werden Objekte aus einem Quell-Bucket in einen oder mehrere Ziel-Bucket(s) repliziert.

Um die Replikation für einen Bucket mit aktivierter Objektsperre einzurichten, können Sie die S3-Konsole, AWS CLI die Amazon S3-REST-API oder AWS SDKs verwenden.

> **Note**
>
> Um die Objektsperre mit der Replikation zu verwenden, müssen Sie zwei zusätzliche Berechtigungen für den S3-Quell-Bucket in der AWS Identity and Access Management (IAM)-Rolle erteilen, die Sie zum Einrichten der Replikation verwenden. Die zwei neuen Berechtigungen sind `s3:GetObjectRetention` und `s3:GetObjectLegalHold`. Wenn die Rolle über eine `s3:Get*`-Berechtigungsanweisung verfügt, ist die Anforderung dadurch erfüllt. Weitere Informationen finden Sie unter Einrichten von Berechtigungen.
> Allgemeine Informationen zur S3-Replikation finden Sie unter Replizieren von Objekten.
> Beispiele für die Einrichtung der S3-Replikation finden Sie unter Anleitungen: Beispiele zum Konfigurieren der Replikation.

Verwenden von Object Lock mit Amazon S3 Inventory

Sie können Amazon S3 Inventory so konfigurieren, dass Listen der Objekte in einem S3-Bucket nach einem definierten Zeitplan erstellt werden. Sie können Amazon S3 Inventory so konfigurieren, dass es die folgenden Object-Lock-Metadaten für Ihre Objekte enthält:

- Das Aufbewahrungsdatum
- Der Aubewahrungsmodus
- Die gesetzliche Aufbewahrungsfrist

Weitere Informationen finden Sie unter Amazon S3 Inventory.

Verwalten von S3-Lebenszyklusrichtlinien mit Object Lock

Konfigurationen für die Verwaltung des Objektlebenszyklus funktionieren für geschützte Objekte weiterhin normal. Dies schließt die Platzierung von Löschmarkierungen ein. Eine gesperrte Version eines Objekts kann jedoch nicht durch eine S3-Lebenszyklus-Ablaufrichtlinie gelöscht werden. Die Objektsperre wird unabhängig davon beibehalten, in welcher Speicherklasse sich das Objekt befindet und während der Übergänge des S3-Lebenszyklus zwischen Speicherklassen.

Weitere Informationen zur Verwaltung von Objektlebenszyklen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Verwalten von Löschmarkierungen mit Object Lock

Sie können eine geschützte Objektversion zwar nicht löschen, aber eine Löschmarkierung für das betreffende Objekt erstellen. Durch das Setzen einer Löschmarkierung auf einem Objekt wird keine Objektversion gelöscht. Amazon S3 verhält sich dadurch aber zumeist so, als ob das Objekt gelöscht worden wäre. Weitere Informationen finden Sie unter Arbeiten mit Löschmarkierungen.

> **Note**
>
> Löschmarkierungen sind nicht WORM-geschützt, unabhängig vom Aufbewahrungszeitraum oder der rechtlichen Aufbewahrungsfrist für das zugrunde liegende Objekt.

Verwenden von S3 Storage Lens mit Object Lock

Wwnn Sie Metriken für objektsperrefähige Speicherbytes und die Anzahl der Objekte sehen möchten, können Sie Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können.

Weitere Informationen finden Sie unter Verwenden von S3 Storage Lens zum Schutz Ihrer Daten.

Eine vollständige Liste der Metriken finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Hochladen von Objekten in einen Bucket mit aktivierter Objektsperre

Der `Content-MD5` Header ist für jede Anforderung zum Hochladen eines Objekts mit einem Aufbewahrungszeitraum erforderlich, der mit Object Lock konfiguriert wurde. Der MD5-Digest ist eine Möglichkeit, die Integrität Ihres Objekts nach dem Hochladen in einen Bucket zu überprüfen. Nach dem Hochladen des Objekts berechnet Amazon S3 den MD5-Digest des Objekts und vergleicht ihn mit dem von Ihnen angegebenen Wert. Die Anforderung ist nur erfolgreich, wenn die beiden Digests übereinstimmen. Die S3-Konsole fügt diesen Header automatisch hinzu. Sie müssen diesen Header jedoch angeben, wenn Sie die PutObject API verwenden.

Weitere Informationen finden Sie unter Verwenden von Content-MD5 beim Hochladen von Objekten.

Konfigurieren von Ereignissen und Benachrichtigungen

Sie können Amazon S3-Ereignisbenachrichtigungen verwenden, um den Zugriff auf Ihre Objektsperrenkonfigurationen und -daten zu verfolgen, indem Sie verwenden AWS CloudTrail.

Weitere Informationen zu CloudTrailfinden Sie unter Was ist AWS CloudTrail? im AWS CloudTrail - Benutzerhandbuch.

Sie können Amazon auch verwenden CloudWatch , um Warnungen basierend auf diesen Daten zu generieren. Weitere Informationen zu CloudWatchfinden Sie unter Was ist Amazon CloudWatch? im Amazon- CloudWatch Benutzerhandbuch.

Festlegen von Beschränkungen für Aufbewahrungsfristen mit einer Bucket-Richtlinie

Sie können mittels einer Bucket-Richtlinie mindestens erforderliche und maximal zulässige Aufbewahrungszeiträume für einen Bucket festlegen. Die maximale Aufbewahrungsfrist beträgt 100 Jahre.

Das folgende Beispiel zeigt eine Bucket-Richtlinie, die den Bedingungsschlüssel `s3:object-lock-remaining-retention-days` verwendet, um einen maximalen Aufbewahrungszeitraum von 10 Tagen festzulegen.

```
{
    "Version": "2012-10-17",
    "Id": "SetRetentionLimits",
    "Statement": [
        {
            "Sid": "SetRetentionPeriod",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "s3:PutObjectRetention"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
            "Condition": {
                "NumericGreaterThan": {
                    "s3:object-lock-remaining-retention-days": "10"
                }
            }
        }
    ]
}
```

> **Note**
>
> Wenn es sich bei Ihrem Bucket um den Ziel-Bucket einer Replikationsrichtlinie handelt, könnenSie minimal und maximal zulässige Aufbewahrungszeiträume für Objektreplikate

einrichten möchten, die mittels Replikation erstellt werden. Hierzu müssen Sie die `s3:ReplicateObject`-Aktion in Ihrer Bucket-Richtlinie zulassen. Weitere Informationen zur Verwendung von Replikationsberechtigungen finden Sie unter the section called “Einrichten von Berechtigungen”.

Weitere Informationen zu Bucket-Richtlinien finden Sie in den folgenden Themen:

- Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz
- Objektoperationen
- Beispiele für Amazon-S3-Bedingungsschlüssel

Konfigurieren von S3 Object Lock

Mit Amazon S3 Object Lock können Sie Objekte in Amazon S3 speichern, indem Sie ein write-once-read-many (WORM)-Modell verwenden. Mit der S3-Objektsperre können Sie für einen festen Zeitraum oder auf unbegrenzte Zeit verhindern, dass ein Objekt gelöscht oder überschrieben wird. Allgemeine Informationen zu den Funktionen von Object Lock finden Sie unter Verwenden der S3-Objektsperre.

Bevor Sie Objekte sperren können, müssen Sie die S3-Versionsverwaltung und Object Lock für einen Bucket aktivieren. Anschließend können Sie einen Aufbewahrungszeitraum, eine gesetzliche Aufbewahrungsfrist oder beides festlegen.

Um mit Object Lock zu arbeiten, müssen Sie über bestimmte Berechtigungen verfügen. Eine Liste der Berechtigungen für verschiedene Object-Lock-Vorgänge finden Sie unter the section called “Erforderliche Berechtigungen”.

Important

- Wenn Sie Object Lock für einen Bucket aktiviert haben, können Sie Object Lock nicht deaktivieren oder die Versionsverwaltung für diesen Bucket aussetzen.
- S3-Buckets mit Object Lock können nicht als Ziel-Buckets für Server-Zugriffsprotokolle verwendet werden. Weitere Informationen finden Sie unter the section called “Protokollierungs-Serverzugriff”.

Themen

Aktivieren von Object Lock beim Erstellen eines neuen S3-Buckets

Sie können die Objektsperre aktivieren, wenn Sie einen neuen S3-Bucket mithilfe der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI), der AWS SDKs oder der Amazon S3-REST-API erstellen.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie Create Bucket (Bucket erstellen) aus.

   Anschließend wird die Seite Bucket erstellen geöffnet.
4. Geben Sie unter Bucket Name (Bucket-Name) einen Namen für den Bucket ein.

   > **Note**
   >
   > Der Name eines einmal erstellten Buckets kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.
5. Wählen Sie für Region die aus, AWS-Region in der sich der Bucket befinden soll.
6. Wählen Sie unter Objekt-Eigentümerschaft eine der folgenden Einstellungen aus, um Zugriffssteuerungslisten (ACLs) zu deaktivieren oder zu aktivieren und die Eigentümerschaft von Objekten zu steuern, die in Ihren Bucket hochgeladen wurden:
7. Wählen Sie unter Einstellungen "Öffentlichen Zugriff beschränken" für diesen Bucket die Einstellungen zum Beschränken des öffentlichen Zugriffs aus, die Sie auf den Bucket anwenden möchten.

8. Wählen Sie unter Bucket-Versionsverwaltung die Option Aktiviert.

   Object Lock funktioniert nur mit versionsgesteuerten Buckets.

9. (Optional) Unter Tags können Sie auswählen, ob Sie Ihrem Bucket Tags hinzufügen möchten. Tags sind Schlüssel-Wert-Paare, die zur Kategorisierung von Speicher und zur Zuweisung von Kosten verwendet werden.

10. Suchen Sie unter Erweiterte Einstellungen nach Object Lock und wählen Sie Aktivieren aus.

    Sie müssen sich darüber im Klaren sein, dass durch die Aktivierung von Object Lock Objekte in diesem Bucket dauerhaft gesperrt werden können.

11. Wählen Sie Bucket erstellen aus.

Verwenden der AWS CLI

Im folgenden `create-bucket`-Beispiel wird ein neuer S3-Bucket mit dem Namen *DOC-EXAMPLE-BUCKET1* und aktiviertem Object Lock erstellt:

```
aws s3api create-bucket --bucket DOC-EXAMPLE-BUCKET1 --object-lock-enabled-for-bucket
```

Weitere Informationen und Beispiele finden Sie unter create-bucket in der AWS CLI -Befehlsreferenz.

> **Note**
>
> Sie können AWS CLI Befehle über die Konsole ausführen, indem Sie verwenden AWS CloudShell. AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt über die starten können AWS Management Console. Weitere Informationen finden Sie unter Was ist CloudShell? im AWS CloudShell -Benutzerhandbuch.

Verwenden der REST-API

Sie können die REST-API verwenden, um einen neuen S3-Bucket mit aktiviertem Object Lock zu erstellen. Weitere Informationen finden Sie unter CreateBucket in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für die Aktivierung der Objektsperre beim Erstellen eines neuen S3-Buckets mit den - AWS SDKs finden Sie unter Erstellen eines Amazon S3-Buckets mit einem AWS -SDK.

Beispiele für das Abrufen der aktuellen Objektsperrenkonfiguration mit den - AWS SDKs finden Sie unter Abrufen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Aktivieren von Object Lock für einen vorhandenen S3-Bucket

Sie können die Objektsperre für einen vorhandenen S3-Bucket mithilfe der Amazon S3-Konsole, der AWS CLI, AWS SDKs oder der Amazon S3-REST-API aktivieren.

Verwenden der S3-Konsole

> **Note**
>
> Object Lock funktioniert nur mit versionsgesteuerten Buckets.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie Object Lock aktivieren möchten.
4. Wählen Sie die Registerkarte Eigenschaften aus.
5. Scrollen Sie unter Eigenschaften nach unten zum Bereich Object Lock und wählen Sie Bearbeiten aus.
6. Wählen Sie unter Object Lock die Option Aktivieren aus.

   Sie müssen sich darüber im Klaren sein, dass durch die Aktivierung von Object Lock Objekte in diesem Bucket dauerhaft gesperrt werden können.
7. Wählen Sie Änderungen speichern aus.

Verwenden der AWS CLI

Mit dem folgenden `put-object-lock-configuration`-Beispielbefehl wird eine Aufbewahrungsfrist von 50 Tagen für Object Lock für einen Bucket mit dem Namen *`DOC-EXAMPLE-BUCKET1`* festgelegt:

```
aws s3api put-object-lock-configuration --bucket DOC-EXAMPLE-BUCKET1 --object-lock-
configuration='{ "ObjectLockEnabled": "Enabled", "Rule": { "DefaultRetention":
 { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

Weitere Informationen und Beispiele finden Sie unter put-object-lock-configuration in der AWS CLI - Befehlsreferenz.

> **Note**
>
> Sie können AWS CLI Befehle über die Konsole ausführen, indem Sie verwenden AWS CloudShell. AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt über die starten können AWS Management Console. Weitere Informationen finden Sie unter Was ist CloudShell? im AWS CloudShell -Benutzerhandbuch.

Verwenden der REST-API

Sie können die Amazon-S3-REST-API verwenden, um Object Lock für einen vorhandenen S3-Bucket zu aktivieren. Weitere Informationen finden Sie unter PutObjectLockConfiguration in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für die Aktivierung der Objektsperre für einen vorhandenen S3-Bucket mit den - AWS SDKs finden Sie unter Festlegen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK.

Beispiele für das Abrufen der aktuellen Objektsperrenkonfiguration mit den - AWS SDKs finden Sie unter Abrufen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Einrichten oder Ändern einer gesetzlichen Aufbewahrungsfrist für ein S3-Objekt

Sie können eine rechtliche Aufbewahrungsfrist für ein S3-Objekt mithilfe der Amazon S3-Konsole, der - AWS CLI SD AWS SDKs oder der Amazon S3-REST-API festlegen oder entfernen.

> **Important**
>
> - Wenn Sie eine gesetzliche Aufbewahrungsfrist für ein Objekt festlegen möchten, muss Object Lock für den Bucket des Objekts bereits aktiviert sein.
> - Wenn Sie PUT für eine Objektversion mit einem expliziten individuellen Aufbewahrungsmodus und Zeitraum in einem Bucket verwenden, haben die individuellen Object-Lock-Einstellungen der Objektversion Vorrang vor allen Aufbewahrungseinstellungen für Bucket-Eigenschaften.

Weitere Informationen finden Sie unter the section called “Rechtliche Aufbewahrungsfristen”.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält, für das Sie eine gesetzliche Aufbewahrungsfrist einrichten oder ändern möchten.
4. Wähle Sie in der Liste Objekte das Objekt aus, für das Sie eine gesetzliche Aufbewahrungsfrist festlegen oder ändern möchten.
5. Suchen Sie auf der Seite mit den Objekteigenschaften den Bereich Object Lock – Gesetzliche Aufbewahrungsfristen und wählen Sie Bearbeiten.
6. Wählen Sie Aktivieren, um eine gesetzliche Aufbewahrungsfrist festzulegen, oder Deaktivieren, um eine gesetzliche Aufbewahrungsfrist zu entfernen.
7. Wählen Sie Änderungen speichern aus.

Verwenden der AWS CLI

Im folgenden `put-object-legal-hold`-Beispiel wird für das Objekt *`my-image.fs`* im Bucket mit dem Namen *`DOC-EXAMPLE-BUCKET1`* eine gesetzliche Aufbewahrungsfrist eingerichtet:

```
aws s3api put-object-legal-hold --bucket DOC-EXAMPLE-BUCKET1 --key my-image.fs --legal-hold="Status=ON"
```

Im folgenden `put-object-legal-hold`-Beispiel wird für das Objekt *`my-image.fs`* im Bucket mit dem Namen *`DOC-EXAMPLE-BUCKET1`* eine gesetzliche Aufbewahrungsfrist entfernt:

```
aws s3api put-object-legal-hold --bucket DOC-EXAMPLE-BUCKET1 --key my-image.fs --legal-hold="Status=OFF"
```

Weitere Informationen und Beispiele finden Sie unter put-object-legal-hold in der AWS CLI - Befehlsreferenz.

> **Note**
>
> Sie können AWS CLI Befehle über die Konsole ausführen, indem Sie verwenden AWS CloudShell. AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt über die starten können AWS Management Console. Weitere Informationen finden Sie unter Was ist CloudShell? im AWS CloudShell -Benutzerhandbuch.

Verwenden der REST-API

Sie können die REST-API verwenden, um einen gesetzlichen Aufbewahrungszeitraum für ein Objekt festzulegen oder zu ändern. Weitere Informationen finden Sie unter PutObjectLegalHold in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für das Festlegen einer rechtlichen Aufbewahrungsfrist für ein Objekt mit den - AWS SDKs finden Sie unter Festlegen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK.

Beispiele dafür, wie Sie den aktuellen Status der gesetzlichen Aufbewahrungsfrist mit den - AWS SDKs abrufen, finden Sie unter Abrufen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Einrichten oder Ändern eines Aufbewahrungszeitraums für ein S3-Objekt

Sie können einen Aufbewahrungszeitraum für ein S3-Objekt mithilfe der Amazon S3-Konsole, der AWS CLI- AWS SDKs oder der Amazon S3-REST-API festlegen oder ändern.

Important

- Wenn Sie einen Aufbewahrungszeitraum für ein Objekt festlegen möchten, muss Object Lock für den Bucket des Objekts bereits aktiviert sein.
- Wenn Sie PUT für eine Objektversion mit einem expliziten individuellen Aufbewahrungsmodus und Zeitraum in einem Bucket verwenden, haben die individuellen Object-Lock-Einstellungen der Objektversion Vorrang vor allen Aufbewahrungseinstellungen für Bucket-Eigenschaften.
- Die einzige Möglichkeit, ein Objekt im Compliance-Modus vor Ablauf des Aufbewahrungsdatums zu löschen, besteht darin, das zugehörige zu löschen AWS-Konto.

Weitere Informationen finden Sie unter Aufbewahrungszeiträume.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält, für das Sie einen Aufbewahrungszeitraum einrichten oder ändern möchten.
4. Wähle Sie in der Liste Objekte das Objekt aus, für das Sie einen Aufbewahrungszeitraum festlegen oder ändern möchten.
5. Suchen Sie auf der Seite mit den Objekteigenschaften den Bereich Object Lock – Aufbewahrungszeitraum und wählen Sie Bearbeiten.
6. Wählen Sie unter Aufbewahrung die Option Aktivieren aus, um einen Aufbewahrungszeitraum festzulegen, oder Deaktivieren, um einen Aufbewahrungszeitraum zu entfernen.
7. Wenn Sie Aktivieren unter Aufbewahrungsmodus ausgewählt haben, wählen Sie entweder Governance-Modus oder Compliance-Modus. Weitere Informationen finden Sie unter Aufbewahrungsmodi.
8. Wählen Sie unter Aufbewahren bis das Datum aus, an dem der Aufbewahrungszeitraum enden soll. Während dieses Zeitraums ist Ihr Objekt WORM-geschützt und kann weder überschrieben noch gelöscht werden. Weitere Informationen finden Sie unter Aufbewahrungszeiträume.
9. Wählen Sie Save Changes (Änderungen speichern).

Verwenden der AWS CLI

Im folgenden `put-object-retention`-Beispiel wird für das Objekt *`my-image.fs`* in dem Bucket mit dem Namen *`DOC-EXAMPLE-BUCKET1`* ein Aufbewahrungszeitraum bis zum 1. Januar 2025 festgelegt:

```
aws s3api put-object-retention --bucket DOC-EXAMPLE-BUCKET1 --key my-image.fs --
retention='{ "Mode": "GOVERNANCE", "RetainUntilDate": "2025-01-01T00:00:00" }'
```

Weitere Informationen und Beispiele finden Sie unter put-object-retention in der AWS CLI - Befehlsreferenz.

> **Note**
>
> Sie können AWS CLI Befehle über die Konsole ausführen, indem Sie verwenden AWS CloudShell. AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt über die starten können AWS Management Console. Weitere Informationen finden Sie unter Was ist CloudShell? im AWS CloudShell -Benutzerhandbuch.

Verwenden der REST-API

Sie können die REST-API verwenden, um einen Aufbewahrungszeitraum für ein Objekt festzulegen. Weitere Informationen finden Sie unter PutObjectRetention in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für das Festlegen eines Aufbewahrungszeitraums für ein Objekt mit den - AWS SDKs finden Sie unter Festlegen des Aufbewahrungszeitraums eines Amazon S3-Objekts mithilfe eines - AWS SDK.

Beispiele für das Abrufen des Aufbewahrungszeitraums für ein Objekt mit den - AWS SDKs finden Sie unter Abrufen der Aufbewahrungskonfiguration eines Amazon S3-Objekts mithilfe eines - AWS SDK.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Einrichten oder Ändern eines Standard-Aufbewahrungszeitraums für einen S3-Bucket

Sie können einen Standardaufbewahrungszeitraum für einen S3-Bucket festlegen oder ändern, indem Sie die Amazon S3-Konsole AWS CLI, , AWS SDKs oder die Amazon S3-REST-API verwenden. Sie geben eine Dauer in Tagen oder Jahren an, für die jede in dem Bucket befindliche Objektversion geschützt werden soll.

Important

- Wenn Sie einen Standard-Aufbewahrungszeitraum für einen Bucket festlegen möchten, muss Object Lock für den Bucket bereits aktiviert sein.
- Wenn Sie für eine Objektversion mit einem expliziten individuellen Aufbewahrungsmodus und Zeitraum in einem Bucket PUT durchführen, haben die individuellen Object-Lock-Einstellungen der Objektversion Vorrang vor allen Aufbewahrungseinstellungen für Bucket-Eigenschaften.
- Die einzige Möglichkeit, ein Objekt im Compliance-Modus vor Ablauf des Aufbewahrungsdatums zu löschen, besteht darin, das zugehörige zu löschen AWS-Konto.

Weitere Informationen finden Sie unter Aufbewahrungszeiträume.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie einen Aufbewahrungszeitraum einrichten oder ändern möchten.
4. Wählen Sie die Registerkarte Eigenschaften aus.
5. Scrollen Sie unter Eigenschaften nach unten zum Bereich Object Lock und wählen Sie Bearbeiten aus.
6. Wählen Sie unter Standard-Aufbewahrung die Option Aktivieren aus, um einen Standard-Aufbewahrungszeitraum festzulegen, oder Deaktivieren, um einen Standard-Aufbewahrungszeitraum zu entfernen.

7. Wenn Sie Aktivieren unter Aufbewahrungsmodus ausgewählt haben, wählen Sie entweder Governance-Modus oder Compliance-Modus. Weitere Informationen finden Sie unter Aufbewahrungsmodi.
8. Wählen Sie unter Standard-Aufbewahrungszeitraum die Anzahl der Tage oder Jahre aus, für die der Aufbewahrungszeitraum gelten soll. Objekte, die sich in diesem Bucket befinden, werden für diese Anzahl von Tagen oder Jahren gesperrt. Weitere Informationen finden Sie unter Aufbewahrungszeiträume.
9. Wählen Sie Save Changes (Änderungen speichern).

Verwenden der AWS CLI

Mit dem folgenden `put-object-lock-configuration`-Beispielbefehl wird eine Object-Lock-Aufbewahrungsfrist von 50 Tag für einen Bucket mit dem Namen *DOC-EXAMPLE-BUCKET1* unter Verwendung des Compliance-Modus festgelegt:

```
aws s3api put-object-lock-configuration --bucket DOC-EXAMPLE-BUCKET1 --object-lock-
configuration='{ "ObjectLockEnabled": "Enabled", "Rule": { "DefaultRetention":
 { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

Im folgenden `put-object-lock-configuration`-Beispiel wird die Standard-Aufbewahrungskonfiguration für einen Bucket entfernt:

```
aws s3api put-object-lock-configuration --bucket DOC-EXAMPLE-BUCKET1 --object-lock-
configuration='{ "ObjectLockEnabled": "Enabled"}'
```

Weitere Informationen und Beispiele finden Sie unter put-object-lock-configuration in der AWS CLI -Befehlsreferenz.

> **Note**
>
> Sie können AWS CLI Befehle über die Konsole ausführen, indem Sie verwenden AWS CloudShell. AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt über die starten können AWS Management Console. Weitere Informationen finden Sie unter Was ist CloudShell? im AWS CloudShell -Benutzerhandbuch.

Verwenden der REST-API

Sie können die REST-API verwenden, um einen Standardaufbewahrungszeitraum für einen vorhandenen S3-Bucket festzulegen. Weitere Informationen finden Sie unter PutObjectLockConfiguration in der API-Referenz zu Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für das Festlegen eines Standardaufbewahrungszeitraums für einen vorhandenen S3-Bucket mit den - AWS SDKs finden Sie unter Festlegen des Standardaufbewahrungszeitraums eines Amazon S3-Buckets mithilfe eines - AWS SDK.

Allgemeine Informationen zur Verwendung verschiedener AWS SDKs finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.

Verwenden von Amazon-S3-Speicherklassen

Jedem Objekt in Amazon S3 ist eine Speicherklasse zugeordnet. Wenn Sie beispielsweise alle Objekte in einem S3-Bucket auflisten, zeigt die Konsole die Speicherklasse für alle Objekte in der Liste an. Amazon S3 bietet eine Reihe von Speicherklassen für die von Ihnen gespeicherten Objekte an. Sie wählen eine Speicherklasse abhängig von Ihrem Anwendungsszenario und den Leistungsanforderungen für den Zugriff aus. Alle diese Speicherklassen bieten höchste Haltbarkeit.

Die folgenden Abschnitte enthalten Details zu den verschiedenen Speicherklassen und zum Festlegen einer Speicherklasse für Ihre Objekte.

Themen

Speicherklassen für Objekte mit häufigem Zugriff

Für leistungssensible Anwendungsfälle (die eine Zugriffszeit von Millisekunden erfordern) und Daten mit häufigem Zugriff stellt Amazon S3 die folgenden Speicherklassen bereit:

- S3 Standard – Die Standardspeicherklasse. Wenn Sie beim Upload eines Objekts keine Speicherklasse angeben, weist Amazon S3 die Speicherklasse S3 Standard zu.
- S3 Express One Zone – Amazon S3 Express One Zone ist eine hochleistungsfähige Amazon-S3-Speicherklasse mit einer einzelnen Zone, die entwickelt wurde, um einen konsistenten Datenzugriff im einstelligen Millisekundenbereich für Ihre Daten und für latenzempfindliche Anwendungen zu gewährleisten. S3 Express One Zone ist die heute verfügbare Cloud-Objektspeicherklasse mit der niedrigsten Latenz, mit einer bis zu zehnmal schnelleren Datenzugriffsgeschwindigkeit und mit Anforderungskosten, die 50 Prozent niedriger sind als S3 Standard. Mit S3 Express One Zone werden Ihre Daten redundant auf mehreren Geräten innerhalb einer einzigen Availability Zone gespeichert. Weitere Informationen finden Sie unter Was ist S3 Express One Zone?.
- Reduced Redundancy – Die Speicherklasse Reduced Redundancy Storage (RRS) ist für nicht kritische, reproduzierbare Daten vorgesehen, die mit einer Redundanz gespeichert werden können, die unter dem Wert der Speicherklasse S3 Standard liegt.

  > **Important**
  >
  > Wir empfehlen, diese Speicherklasse nicht zu verwenden. Die Speicherklasse S3 Standard ist kostengünstiger.

  In Bezug auf die Zuverlässigkeit gilt für RRS-Objekte ein jährlich zu erwartender Objektverlust von 0,01 %. Geht ein RRS-Objekt verloren, gibt Amazon S3 einen 405-Fehler bei Anforderungen für dieses Objekt zurück.

Speicherklasse zur automatischen Optimierung von Daten mit sich ändernden oder unbekannten Zugriffsmustern

S3 Intelligent-Tiering ist eine Amazon-S3-Speicherklasse zur Optimierung der Speicherkosten durch automatisches Verschieben von Daten auf die kostengünstigste Zugriffsebene ohne Leistungsauswirkung oder Betriebsaufwand. S3 Intelligent-Tiering ist die einzige Cloud-Speicherklasse, die automatische Kosteneinsparungen durch Verschieben von Daten auf granularer

Objektebene zwischen den Zugriffsebenen ermöglicht, wenn sich die Zugriffsmuster ändern. S3 Intelligent-Tiering ist die ideale Speicherklasse, wenn Sie die Speicherkosten für Daten mit unbekannten oder sich ändernden Zugriffsmustern optimieren möchten. Bei S3 Intelligent-Tiering werden keine Abrufgebühren erhoben.

Gegen eine geringe monatliche Gebühr für Objektüberwachung und Automatisierung überwacht S3 Intelligent-Tiering die Zugriffsmuster und verschiebt die Objekte, auf die nicht zugegriffen wurde, automatisch in kostengünstigere Zugriffsebenen. S3 Intelligent-Tiering ermöglicht automatische Speicherkosteneinsparungen in drei Zugriffsebenen mit niedriger Latenz und hohem Durchsatz. Für Daten, auf die asynchron zugegriffen werden kann, können Sie innerhalb der S3 Intelligent-Tiering-Speicherklasse die automatische Archivfunktionen aktivieren. S3 Intelligent-Tiering ist auf 99,9 % Verfügbarkeit und 99,9999999 % Haltbarkeit ausgelegt.

S3 Intelligent-Tiering speichert Objekte automatisch in drei Zugriffsebenen:

- Frequent Access (Häufige Zugriffe) – In S3 Intelligent-Tiering hochgeladene oder übertragene Objekte werden automatisch in der Ebene Frequent Access gespeichert.
- Infrequent Access (Seltene Zugriffe) – S3 Intelligent-Tiering verschiebt Objekte, auf die 30 aufeinanderfolgende Tage lang nicht zugegriffen wurde, in die Ebene Infrequent Access.
- Archive Instant Access (Archiv für sofortigen Zugriff) – Mit S3 Intelligent-Tiering werden alle vorhandenen Objekte, auf die 90 aufeinanderfolgende Tage lang nicht zugegriffen wurde, automatisch in die Ebene Archive Instant Access verschoben.

Zusätzlich zu diesen drei Stufen bietet S3 Intelligent-Tiering zwei optionale Archivzugriffsebenen:

- Archive Access (Archivzugriff) – S3 Intelligent-Tiering bietet Ihnen die Möglichkeit, die Archivzugriffsebene für Daten zu aktivieren, auf die asynchron zugegriffen werden kann. Nach der Aktivierung archiviert die Ebene Archive Access automatisch Objekte, auf die mindestens 90 aufeinanderfolgende Tage lang nicht zugegriffen wurde.
- Deep Archive Access (tiefer Archivzugriff) – S3 Intelligent-Tiering bietet Ihnen die Möglichkeit, die Stufe Deep Archive Access für Daten zu aktivieren, auf die asynchron zugegriffen werden kann. Nach der Aktivierung archiviert die Ebene Deep Archive Access automatisch Objekte, auf die mindestens 180 aufeinanderfolgende Tage lang nicht zugegriffen wurde.

> **Note**
>
> - Aktivieren Sie die Stufe Archive Access nur für 90 Tage, wenn Sie die Stufe Instant Access Archiv umgehen möchten. Die Stufe Archive Access bietet etwas niedrigere Speicherkosten mit minute-to-hour Abrufzeiten. Die Stufe Archive Instant Access bietet Millisekunden-Zugriff und hohe Durchsatzleistung.
> - Aktivieren Sie die Ebenen Archivzugriff und Deep Archive Access nur, wenn Ihre Anwendung asynchron auf Ihre Objekte zugreifen kann. Wenn das Objekt, das Sie abrufen, in den Stufen Archive Access oder Deep Archive Access gespeichert ist, stellen Sie das Objekt zunächst mithilfe von `RestoreObject` wieder her.

Sie können neu erstellte Daten in S3 Intelligent-Tiering verschieben und dies als Standardspeicherklasse festlegen. Sie können auch eine oder beide Archivzugriffsebenen mithilfe der `PutBucketIntelligentTieringConfiguration`-API-Operation AWS CLI, der oder der Amazon S3-Konsole aktivieren. Weitere Informationen zur Verwendung von S3 Intelligent-Tiering und zur Aktivierung der Archivzugriffsebenen finden Sie unter Verwenden von S3 Intelligent-Tiering.

Um auf Objekte in den Stufen „Archive Access“ oder „Deep Archive Access“ zugreifen zu können, müssen Sie sie zunächst wiederherstellen. Weitere Informationen finden Sie unter Wiederherstellen von Objekten aus den Stufen S3 Intelligent-Tiering Archive Access und Deep Archive Access.

> **Note**
>
> Wenn ein Objekt kleiner als 128 KB ist, wird es nicht überwacht und ist nicht für das Auto-Tiering geeignet. Kleinere Objekte werden immer in der Stufe für häufige Zugriffe gespeichert. Weitere Informationen zu S3 Intelligent-Tiering finden Sie unter S3-Intelligent-Tiering-Zugriffsebenen.

Speicherklassen für Objekte mit seltenem Zugriff

Die Speicherklassen S3 Standard-IA und S3 One Zone-IA sind für langlebige Daten, auf die selten zugegriffen wird, konzipiert. (IA steht für Infrequent Access (seltener Zugriff).) S3 Standard-IA- und S3 One Zone-IA-Objekte sind für den Zugriff in Millisekunden verfügbar (ähnlich wie die Speicherklasse S3 Standard). Da Amazon S3 eine Abrufgebühr für diese Objekte erhebt, sind sie am besten für

Daten mit seltenem Zugriff geeignet. Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

Beispielsweise können Sie die Speicherklassen S3 Standard-IA und S3 One Zone-IA für Folgendes auswählen:

- Zum Speichern von Sicherungen.
- Für ältere Daten, auf die selten zugegriffen wird, für die aber dennoch ein Zugriff in Millisekunden erforderlich ist. Beim Upload von Daten können Sie beispielsweise die S3 Standard-Speicherklasse auswählen und dann Amazon S3 mithilfe der Lebenszyklus-Konfiguration anweisen, die Objekte in die Klasse S3 Standard-IA oder S3 One Zone-IA zu überführen.

  Weitere Informationen zur Lebenszyklusverwaltung finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

> **Note**
>
> Die Speicherklassen S3 Standard-IA und S3 One Zone-IA sind für Objekte mit mehr als 128 KB geeignet, die Sie für mindestens 30 Tage aufbewahren wollen. Ist ein Objekt kleiner als 128 KB, fallen Amazon-S3-Gebühren für 128 KB an. Wenn Sie ein Objekt vor dem Ablauf der 30-tägigen Mindestspeicherdauer löschen, werden Ihnen 30 Tage berechnet. Für Objekte, die vor Ablauf der 30-Tage-Frist gelöscht, überschrieben oder in eine andere Speicherklasse verschoben werden, wird die übliche nutzungsabhängige Gebühr sowie eine anteilige Gebühr für die restlichen Tage des 30-Tage-Minimums in Rechnung gestellt. Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

Diese Speicherklassen weisen folgende Unterschiede auf:

- S3 Standard-IA – Amazon S3 speichert die Objektdaten redundant in mehreren geografisch getrennten Availability Zones (ähnlich wie die Speicherklasse S3 Standard). S3 Standard-IA-Objekte sind von einem Availability Zone-Ausfall nicht betroffen. Diese Speicherklasse bietet mehr Verfügbarkeit und Stabilität als die Klasse S3 One Zone-IA.
- S3 One Zone-IA – Amazon S3 speichert die Objektdaten nur in einer Availability Zone, daher ist diese Variante kostengünstiger als S3 Standard-IA. Allerdings sind die Daten bei einem physischen Ausfall der Availability Zone (aufgrund von Katastrophen wie z. B. Erdbeben und Überflutungen) nicht sicher. Die Speicherklasse S3 One Zone-IA bietet dieselbe Zuverlässigkeit wie S3 Standard-

IA, ist jedoch weniger verfügbar und weniger ausfallsicher. Einen Vergleich der Zuverlässigkeit und Verfügbarkeit von Speicherklassen finden Sie unter Vergleich der Amazon-S3-Speicherklassen am Ende dieses Abschnitts. Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

Wir empfehlen Folgendes:

- S3 Standard-IA – Verwenden Sie diese Klasse für die primäre (oder einzige) Kopie der Daten, die nicht neu erstellt werden können.
- S3 One Zone-IA – Verwenden Sie diese Klasse, wenn Sie die Daten im Falle eines Availability-Zone-Ausfalls neu erstellen können sowie für Objekt-Replicas bei regionsübergreifender Replikation (Cross-Region Replication, CRR) in S3.

Speicherklassen für die Archivierung von Objekten

Die Speicherklassen S3 Glacier Instant Retrieval und S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive dienen zur kostengünstigen Archivierung von Daten. Diese Speicherklassen bieten dieselbe Zuverlässigkeit und Stabilität wie die Speicherklasse S3 Standard und S3 Standard-IA Speicherklassen. Einen Vergleich der Haltbarkeit und Verfügbarkeit der Speicherklassen finden Sie unter Vergleich der Amazon-S3-Speicherklassen.

> **Note**
>
> Wenn Sie S3-Glacier-Speicherklassen verwenden, verbleiben Ihre Objekte in Amazon S3. Sie können nicht direkt über den separaten Amazon-S3-Glacier-Service darauf zugreifen. Weitere Informationen zum Amazon S3-Glacier-Service finden Sie im Entwicklerhandbuch für Amazon S3 Glacier.

Die S3-Glacier-Speicherklassen unterscheiden sich wie folgt:

- S3 Glacier Instant Retrieval – Verwenden Sie diese Klasse zur Archivierung von Daten, auf die nur selten zugegriffen wird und die in Millisekunden abgerufen werden müssen. Daten, die in der Speicherklasse S3 Glacier Instant Retrieval gespeichert sind, bieten eine Kosteneinsparung im Vergleich zur S3-Standard-IA-Speicherklasse mit derselben Latenz- und Durchsatzleistung wie die S3-Standard-IA-Speicherklasse. S3 Glacier Instant Retrieval hat höhere Datenzugriffskosten als S3 Standard-IA.

Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

- S3 Glacier Flexible Retrieval – Diese Klasse eignet sich für Archive, bei denen Teile der Daten möglicherweise innerhalb von wenigen Minuten abgerufen werden müssen. Für in der Speicherklasse S3 Glacier Flexieble Retrieval gespeicherte Daten gilt eine Mindestspeicherdauer von 90 Tagen; mit beschleunigtem Abruf sind die Daten in nur 1–5 Minuten abrufbar. Die Abrufzeit ist flexibel und Sie können in bis zu 5–12 Stunden kostenlose Massenabrufe anfordern. Falls Sie ein Objekt vor dem 90-tägigen Minimum gelöscht, überschrieben oder an eine andere Speicherklasse übertragen haben, werden Ihnen 90 Tage in Rechnung gestellt. Amazon S3 unterstützt Wiederherstellungsanforderungen mit einer Geschwindigkeit von bis zu 1 000 Transaktionen pro Sekunde, pro AWS-Konto für S3 Glacier Flexible Retrieval.

  Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

- S3 Glacier Deep Archive – Verwenden Sie diese Speicherklasse zur Archivierung von Daten, auf die nur selten zugegriffen werden muss. Für in der Speicherklasse S3 Glacier Deep Archive gespeicherte Daten gilt eine Mindestspeicherdauer von 180 Tagen und die Standard-Abrufzeit beträgt 12 Stunden. Falls Sie ein Objekt vor dem 180-tägigen Minimum gelöscht, überschrieben oder an eine andere Speicherklasse übertragen haben, werden Ihnen 180 Tage in Rechnung gestellt. Amazon S3 unterstützt Wiederherstellungsanforderungen mit einer Geschwindigkeit von bis zu 1 000 Transaktionen pro Sekunde, je AWS-Konto für die Speicherklasse S3 Glacier Deep Archive.

  Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

  S3 Glacier Deep Archive ist die kostengünstigste Speicheroption in AWS. Die Speicherkosten für S3 Glacier Deep Archive sind niedriger als für die Speicherklasse S3 Glacier Flexible Retrieval Speicherklasse. Sie können die Abrufkosten für S3 Glacier Deep Archive durch Massenabrufe senken, bei denen Daten innerhalb von 48 Stunden zurückgegeben werden.

Abrufen archivierter Objekte

Sie können die Speicherklasse eines Objekts auf S3 Glacier Flexieble Retrieval oder S3 Glacier Deep Archive setzen, genau so wie für andere Speicherklassen, wie im Abschnitt Einrichten der Speicherklasse eines Objekts beschrieben. Die mit S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive archivierten Objekte sind jedoch nicht für den Echtzeitzugriff verfügbar. Sie müssen zuerst die Objekte von S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive wiederherstellen, bevor Sie darauf zugreifen können. (Objekte von S3 Standard, Reduced Redundancy Storage (RRS), S3

Standard-IA, S3 One Zone-IA, S3 Glacier Instant Retrieval und S3 Intelligent Tiering stehen jederzeit für den Zugriff zur Verfügung.) Weitere Informationen zum Abruf archivierter Objekte finden Sie unter Wiederherstellen eines archivierten Objekts.

> **Important**
>
> Wenn Sie die Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive wählen, bleibt Ihr Objekt in Amazon S3. Sie können nicht direkt über den separaten Amazon-S3-Glacier-Service darauf zugreifen.

Erste Schritte mit den S3-Glacier-Speicherklassen

Weitere Informationen zur Verwendung der Amazon-S3-Glacier-Speicherklassen finden Sie unter Tutorial: Erste Schritte mit den Amazon-S3-Glacier-Speicherklassen.

Speicherklasse für Amazon S3 in Outposts

Mit Amazon S3 on Outposts können Sie S3-Buckets auf Ihren AWS Outposts Ressourcen erstellen und Objekte On-Premises für Anwendungen speichern und abrufen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern. Sie können dieselben API-Operationen und Funktionen AWS Outposts in wie in Amazon S3 verwenden, einschließlich Zugriffsrichtlinien, Verschlüsselung und Tagging. Sie können S3 on Outposts über die AWS Management Console, die AWS CLI-SDKs oder die REST-API verwenden. AWS SDKs

S3 in Outposts bietet eine neue Speicherklasse: S3 Outposts (`OUTPOSTS`). Die Speicherklasse S3 Outposts ist nur für Objekte verfügbar, die in Buckets in Outposts gespeichert sind. Wenn Sie versuchen, diese Speicherklasse mit einem S3-Bucket in einem zu verwenden AWS-Region, tritt ein `InvalidStorageClass` Fehler auf. Sie erhalten dieselbe Fehlermeldung, wenn Sie versuchen, andere S3-Speicherklassen mit Objekten zu verwenden, die in S3-on-Outposts-Buckets gespeichert sind.

Objekte, die in der Speicherklasse S3 Outposts (`OUTPOSTS`) gespeichert sind, werden immer mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) verschlüsselt. Weitere Informationen finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

Sie können auch explizit auswählen, dass Objekte, die in der Speicherklasse S3 Outposts gespeichert sind, mit serverseitiger Verschlüsselung mit vom Kunden bereitgestellten

Verschlüsselungsschlüsseln (SSE-C) verschlüsselt werden. Weitere Informationen finden Sie unter Verwenden von serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C).

> **Note**
>
> S3 on Outposts unterstützt keine serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS).

Weitere Informationen zu S3 on Outposts finden Sie unter Was ist Amazon S3 on Outposts?.

Vergleich der Amazon-S3-Speicherklassen

In der folgenden Tabelle werden die Speicherklassen verglichen, einschließlich ihrer Verfügbarkeit, Haltbarkeit, Mindestspeicherdauer und anderer Faktoren.

Storage Class	Designed for	Durability (designed for)	Availability (designed for)	Availability Zones	Min storage duration	Min billable object size	Other Considerations
STANDARD	Frequently accessed data	99.999999999%	99.99%	>= 3	None	None	None
STANDARD_IA	Long-lived, infrequently accessed data	99.999999999%	99.9%	>= 3	30 days	128 KB	Per GB retrieval fees apply.
INTELLIGENT_TIERING	Long-lived data with changing or unknown access patterns	99.999999999%	99.9%	>= 3	30 days	None	Monitoring and automation fees per object apply. No retrieval fees.
ONEZONE_IA	Long-lived, infrequently accessed, non-critical data	99.999999999%	99.5%	1	30 days	128 KB	Per GB retrieval fees apply. Not resilient to the loss of the Availability Zone.
GLACIER	Long-term data archiving with retrieval times ranging from minutes to hours	99.999999999%	99.99% (after you restore objects)	>= 3	90 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
DEEP_ARCHIVE	Archiving rarely accessed data with a default retrieval time of 12 hours	99.999999999%	99.99% (after you restore objects)	>= 3	180 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
RRS (Not recommended)	Frequently accessed, non-critical data	99.99%	99.99%	>= 3	None	None	None

* S3 Glacier Flexible Retrieval erfordert 40 KB zusätzliche Metadaten für jedes archivierte Objekt. Dazu gehören 32 KB Metadaten, die mit dem Tarif von S3 Glacier Flexible Retrieval abgerechnet werden (erforderlich, um Ihre Daten zu identifizieren und abzurufen), und weitere 8 KB Daten, die zum S3-Standardtarif berechnet werden. Der S3-Standardtarif ist erforderlich, um den benutzerdefinierten Namen und die Metadaten für Objekte zu verwalten, die in S3 Glacier Flexible Retrieval archiviert wurden. Weitere Informationen zu Speicherklassen finden Sie unter Amazon-S3-Speicherklassen.

** S3 Glacier Deep Archive erfordert 40 KB zusätzliche Metadaten für jedes archivierte Objekt. Dazu gehören 32 KB Metadaten, die mit dem Tarif von S3 Glacier Deep Archive abgerechnet werden (erforderlich, um Ihre Daten zu identifizieren und abzurufen), und weitere 8 KB Daten, die zum S3-Standardtarif berechnet werden. Der S3-Standardtarif ist erforderlich, um den benutzerdefinierten Namen und die Metadaten für Objekte zu verwalten, die in Amazon S3 Glacier Deep Archive archiviert wurden. Weitere Informationen zu Speicherklassen finden Sie unter Amazon-S3-Speicherklassen.

Beachten Sie, dass alle Speicherklassen außer S3 One Zone-IA und S3 Express One Zone gegenüber dem physischen Verlust einer Availability Zone aufgrund von Katastrophen ausfallsicher sind. Berücksichtigen Sie neben den Leistungsanforderungen Ihres Anwendungsszenarios auch die Kosten. Preisinformationen für Speicherklassen finden Sie unter Amazon S3 – Preise.

Einrichten der Speicherklasse eines Objekts

Um Objektspeicherklassen festzulegen und zu aktualisieren, können Sie die Amazon S3-Konsole, AWS SDKs oder die AWS Command Line Interface () verwenden AWS CLI. Alle diese Ansätze verwenden Amazon S3-API-Operationen, um Anfragen an Amazon S3 zu senden.

Die API-Operationen von Amazon S3 unterstützen das Einrichten (oder Aktualisieren) von Speicherklassen für Objekte wie folgt:

- Beim Erstellen eines neuen Objekts können Sie dessen Speicherklasse angeben. Wenn Sie beispielsweise neue Objekte mit den API-Operationen PUT Object, POST Object und Initiate Multipart Upload erstellen, fügen Sie zum Spezifizieren einer Speicherklasse den `x-amz-storage-class`-Anforderungs-Header hinzu. Falls Sie diesen Header weglassen, verwendet Amazon S3 die Standardspeicherklasse S3 Standard.
- Sie können auch die Speicherklasse eines bereits in Amazon S3 gespeicherten Objekts in eine beliebige andere Speicherklasse ändern, indem Sie mit der API-Operation PUT Object - Copy eine Kopie des Objekts erstellen. Sie können jedoch nicht PUTT Object - Copy verwenden, um in den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeicherte Objekte zu kopieren. Sie können auch nicht von S3 One Zone-IA zu S3 Glacier Instant Retrieval wechseln.

  Sie kopieren das Objekt unter Verwendung desselben Schlüsselnamens in denselben Bucket und geben die Anforderungs-Header wie folgt an:

  - Legen Sie den `x-amz-metadata-directive`-Header auf `COPY` fest.
  - Legen Sie den `x-amz-storage-class`-Header auf die zu verwendende Speicherklasse fest.

In einem Bucket mit aktiviertem Versioning kann die Speicherklasse einer spezifischen Objektversion nicht geändert werden. Beim Kopieren des Objekts vergibt Amazon S3 eine neue Versions-ID.

- Sie können die Speicherklasse eines Objekts mithilfe der Amazon-S3-Konsole ändern, wenn die Objektgröße weniger als 160 GB beträgt. Bei größeren Objekten empfehlen wir, eine S3-Lebenszykluskonfiguration hinzuzufügen, um die Speicherklasse des Objekts zu ändern.
- Sie können Amazon S3 anweisen, die Speicherklasse von Objekten zu ändern, indem Sie einem Bucket eine Lebenszyklus-Konfiguration hinzufügen. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.
- Bei der Einrichtung der Replikation können Sie eine beliebige andere Speicherklasse für die replizierten Objekte einrichten. Sie können jedoch nicht in den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeicherte Objekte replizieren. Weitere Informationen finden Sie unter Replikations-Konfiguration.

Einschränken von Zugriffsrichtlinienberechtigungen auf eine bestimmte Speicherklasse

Wenn Sie Zugriffsrichtlinienberechtigungen für Amazon-S3-Vorgänge erteilen, können Sie mit dem Bedingungsschlüssel `s3:x-amz-storage-class` einschränken, welche Speicherklasse beim Speichern hochgeladener Objekte verwendet werden soll. Wenn Sie beispielsweise die Berechtigung `s3:PutObject` erteilen, können Sie das Hochladen von Objekten auf eine bestimmte Speicherklasse einschränken. Eine Beispielrichtlinie finden Sie unter Beispiel 5: Objekt-Uploads auf Objekte mit einer bestimmten Speicherklasse beschränken.

Weitere Informationen zur Verwendung von Bedingungen in Richtlinien und die vollständige Liste der Bedingungsschlüssel in Amazon S3 finden Sie in den folgenden Themen:

- Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz
- Beispiele für Amazon-S3-Bedingungsschlüssel

Amazon S3 Intelligent Tiering

Die Speicherklasse S3 Intelligent-Tiering wurde entwickelt, um die Speicherkosten zu optimieren, indem Daten automatisch auf die kostengünstigste Zugriffsebene verschoben werden, wenn sich die Zugriffsmuster ändern, ohne dass Betriebsaufwand oder Leistungseinbußen auftreten. Gegen eine

geringe monatliche Gebühr für Objektüberwachung und Automatisierung überwacht S3 Intelligent-Tiering die Zugriffsmuster und verschiebt die Objekte, auf die nicht zugegriffen wurde, automatisch in kostengünstigere Zugriffsebenen.

S3 Intelligent-Tiering ermöglicht automatische Speicherkosteneinsparungen in drei Zugriffsebenen mit niedriger Latenz und hohem Durchsatz. Für Daten, auf die asynchron zugegriffen werden kann, können Sie innerhalb der S3 Intelligent-Tiering-Speicherklasse die automatische Archivfunktionen aktivieren. Bei S3 Intelligent-Tiering werden keine Abrufgebühren erhoben. Wenn später auf ein Objekt in der Stufe Infrequent Access oder Archive Instant Access zugegriffen wird, wird es automatisch zurück in die Stufe Frequent Access verschoben. Es werden keine Zusatzgebühren erhoben, wenn Objekte innerhalb der S3-Intelligent-Tiering-Speicherklasse zwischen Zugriffsstufen verschoben werden.

S3 Intelligent-Tiering ist die empfohlene Speicherklasse für Daten mit unbekannten, sich ändernden oder unvorhersehbaren Zugriffsmustern, unabhängig von der Objektgröße oder dem Aufbewahrungszeitraum, wie Data Lakes, Datenanalytik und neue Anwendungen.

Weitere Informationen zur Verwendung von S3 Intelligent-Tiering finden Sie in den folgenden Abschnitten:

Themen

So funktioniert S3 Intelligent-Tiering

Die Speicherklasse Amazon S3 Intelligent-Tiering speichert Objekte automatisch in drei Zugriffsebenen. Eine Stufe ist für häufigen Zugriff optimiert, eine kostengünstigere Stufe ist für seltenen Zugriff optimiert und eine andere sehr kostengünstige Stufe ist für selten zugängliche Daten optimiert. Gegen eine geringe monatliche Gebühr für Objektüberwachung und Automatisierung überwacht S3 Intelligent-Tiering die Zugriffsmuster und verschiebt Objekte automatisch in die Ebene für seltenen Zugriff, wenn auf sie 30 aufeinanderfolgende Tage lang nicht zugegriffen wurde. Nach 90 Tagen ohne Zugriff werden die Objekte ohne Leistungseinbußen oder Betriebsaufwand auf die Ebene des Archivs Instant Access verschoben.

Aktivieren Sie Archivierungsfunktionen, um zwei zusätzliche Zugriffsebenen hinzuzufügen und die niedrigsten Speicherkosten für Daten zu erhalten, auf die in Minuten bis Stunden zugegriffen

werden kann. Sie können Objekte schichtweise auf die Archive Access Tier oder Deep Archive Access Tier oder beiden reduzieren. Mit Archive Access verschiebt S3 Intelligent-Tiering Objekte, auf die mindestens 90 aufeinanderfolgende Tage lang nicht zugegriffen wurde, in die Ebene „Archive Access“. Mit Deep Archive Access verschiebt S3 Intelligent-Tiering Objekte, auf die mindestens 180 aufeinanderfolgende Tage lang nicht zugegriffen wurde, in die Ebene „Deep Archive Access“. Für beide Ebenen können Sie die Anzahl der Tage ohne Zugriff je nach Bedarf konfigurieren.

Die folgenden Aktionen stellen einen Zugriff dar, der ein Herabstufen Ihrer Objekte in die Ebene „Archive Access“ oder „Deep Archive Access“ verhindert:

- Herunterladen oder Kopieren eines Objekts über die Amazon S3-Konsole.
- Aufrufen von CopyObject, UploadPartCopy oder Replizieren bestehender Objekte mit S3- Batch-Replikation. In diesen Fällen werden die Quellobjekte der Kopier- oder Replikationsvorgänge hochgestuft.
- Aufrufen von GetObject, PutObject, RestoreObject, CompleteMultipartUpload, ListParts oder SelectObjectContent.

Wenn beispielsweise vor Ablauf der von Ihnen festgelegte Anzahl von Tagen ohne Zugriff (z. B. 180 Tage) über `SelectObjectContent` ein Zugriff auf Ihre Objekte erfolgt, wird durch diese Aktion der Timer zurückgesetzt. Ihre Objekte werden erst in die Ebene „Archive Access“ oder „Deep Archive Access“ verschoben, wenn die Zeit nach der letzten `SelectObjectContent`-Anforderung die von Ihnen festgelegte Anzahl von Tagen erreicht hat.

Wenn später auf ein Objekt in der Stufe Infrequent Access oder Archive Instant Access zugegriffen wird, wird es automatisch zurück in die Stufe Frequent Access verschoben.

Die folgenden Aktionen stellen einen Zugriff dar, bei dem Objekte automatisch von der Ebene „Infrequent Access“ oder „Archive Instant Access“ zurück in die Ebene „Frequent Access“ verschoben werden:

- Herunterladen oder Kopieren eines Objekts über die Amazon S3-Konsole.
- Aufrufen von CopyObject, UploadPartCopy oder Replizieren bestehender Objekte mit Batch-Replikation. In diesen Fällen werden die Quellobjekte der Kopier- oder Replikationsvorgänge hochgestuft.
- Aufrufen von GetObject, PutObject, RestoreObject, CompleteMultipartUpload oder ListParts.

Andere Aktionen stellen keinen Zugriff dar, bei dem Objekte automatisch von der Ebene „Infrequent Access“ oder „Archive Instant Access“ zurück in die Ebene „Frequent Access“ verschoben werden. Im Folgenden finden Sie ein Beispiel, keine endgültige Liste, solcher Aktionen:

- Aufrufen von HeadObject, GetObjectTagging, PutObjectTagging, ListObjects, ListObjectsV2 oder ListObjectVersions.
- Das Aufrufen von SelectObjectContent stellt keinen Zugriff dar, bei dem Objekte zur Ebene „Frequent Access“ hochgestuft werden. Darüber hinaus wird dadurch nicht verhindert, dass Objekte von der Stufe „Frequent Access“ auf die Stufe „Infrequent Access“ und von dort aus auf die Stufe „Archive Instant Access“ herabgestuft werden.

Sie können S3 Intelligent-Tiering als Standardspeicherklasse für neu erstellte Daten konfigurieren, indem Sie `INTELLIGENT-TIERING` in Ihrem PutBucketIntelligentTieringConfiguration-Anforderungs-Header festlegen. S3 Intelligent-Tiering ist auf 99,9 % Verfügbarkeit und 99,9999999 % Haltbarkeit ausgelegt.

> **Note**
>
> Wenn die Größe eines Objekts weniger als 128 KB beträgt, wird es nicht überwacht und ist nicht für das automatische Tiering geeignet. Kleinere Objekte werden immer in der Stufe für häufige Zugriffe gespeichert.

S3-Intelligent-Tiering-Zugriffsebenen

Im folgenden Abschnitt werden die verschiedenen automatischen und optionalen Zugriffsebenen erläutert. Wenn Objekte zwischen Zugriffsebenen verschoben werden, bleibt die Speicherklasse gleich (S3 Intelligent-Tiering).

Stufe für häufige Zugriffe (automatisch)

Hierbei handelt es sich um die Standardzugriffsebene, in der Objekte, die in S3 Intelligent-Tiering erstellt oder überführt sind, seinen Lebenszyklus beginnt. Ein Objekt verbleibt in dieser Ebene, solange darauf zugegriffen wird. Die Stufe für häufige Zugriffe bietet niedrige Latenz und hohe Durchsatzleistung.

Stufe für seltene Zugriffe (automatisch)

Wenn 30 aufeinanderfolgende Tage auf ein Objekt nicht zugegriffen wird, wird das Objekt in die Stufe für seltene Zugriffe verschoben. Die Stufe für seltene Zugriffe bietet niedrige Latenz und hohe Durchsatzleistung.

Stife Archive Instant Access (automatisch)

Wenn 90 aufeinanderfolgende Tage auf ein Objekt nicht zugegriffen wird, wird das Objekt in die Stufe Archive Instant Access verschoben. Die Stufe Archive Instant Access bietet niedrige Latenzzeiten und hohe Durchsatzleistung.

Archivzugriffsebene (optional)

S3 Intelligent-Tiering bietet Ihnen die Möglichkeit, die Archivzugriffsebene für Daten zu aktivieren, auf die asynchron zugegriffen werden kann. Nach der Aktivierung archiviert die Ebene Archive Access automatisch Objekte, auf die mindestens 90 aufeinanderfolgende Tage lang nicht zugegriffen wurde. Sie können die letzte Zugriffszeit für die Archivierung auf maximal 730 Tage verlängern. Die Archivzugriffsebene hat dieselbe Leistung wie die Speicherklasse S3 Glacier Flexible Retrieval.

Die Standardabrufzeiten für diese Zugriffebene können zwischen 3 und 5 Stunden liegen. Wenn Sie Ihre Wiederherstellungsanfrage mithilfe von S3 Batch Operations starten, beginnt Ihre Wiederherstellung innerhalb von Minuten. Weitere Informationen zu den Abrufoptionen finden -Zeiten finden Sie unter the section called “ Wiederherstellen von Objekten aus den Stufen S3 Intelligent-Tiering Archive Access und Deep Archive Access”.

> **Note**
>
> Aktivieren Sie die Stufe Archive Access nur für 90 Tage, wenn Sie die Stufe Instant Access Archiv umgehen möchten. Die Ebene „Archive Access“ liefert etwas niedrigere Speicherkosten mit Abrufzeiten von Minuten zu Stunde. Die Stufe Archive Instant Access bietet Millisekunden-Zugriff und hohe Durchsatzleistung.

Zugriffsebene für Deep Archive (optional)

S3 Intelligent-Tiering bietet Ihnen die Möglichkeit, die Stufe Deep Archive Access für Daten zu aktivieren, auf die asynchron zugegriffen werden kann. Nach der Aktivierung archiviert die Ebene Deep Archive Access automatisch Objekte, auf die mindestens 180 aufeinanderfolgende Tage

lang nicht zugegriffen wurde. Sie können die letzte Zugriffszeit für die Archivierung auf maximal 730 Tage verlängern. Die Deep-Archive-Zugriffsebene hat dieselbe Leistung wie die S3 Glacier Deep Archive-Speicherklasse.

Der Standardabruf von Objekten in dieser Zugriffsebene erfolgt innerhalb von 12 Stunden. Wenn Sie Ihre Wiederherstellungsanfrage mithilfe von S3 Batch Operations starten, beginnt Ihre Wiederherstellung innerhalb von 9 Stunden. Weitere Informationen zu den Abrufoptionen finden -Zeiten finden Sie unter the section called “ Wiederherstellen von Objekten aus den Stufen S3 Intelligent-Tiering Archive Access und Deep Archive Access”.

> **Note**
>
> Aktivieren Sie die Ebenen Archivzugriff und Deep Archive Access nur, wenn Ihre Anwendung asynchron auf Ihre Objekte zugreifen kann. Wenn das Objekt, das Sie abrufen, in den Ebenen „Archive Access“ oder „Deep Archive Access“ gespeichert ist, müssen Sie das Objekt zunächst mithilfe des `RestoreObject`-Vorgangs wieder herstellen.

Verwenden von S3 Intelligent-Tiering

Sie können die Speicherklasse S3 Intelligent-Tiering verwenden, um die Speicherkosten automatisch zu optimieren. S3 Intelligent-Tiering ermöglicht automatische Kosteneinsparungen durch Verschieben von Daten auf granularer Objektebene zwischen den Zugriffsebenen, wenn sich die Zugriffsmuster ändern. Für Daten, auf die asynchron zugegriffen werden kann, können Sie innerhalb der S3 Intelligent-Tiering-Speicherklasse die automatische Archivfunktionen mittels AWS Management Console, AWS CLI, oder Amazon S3 API aktivieren.

Verschieben von Daten nach S3 Intelligent-Tiering

Es gibt zwei Möglichkeiten, Daten in S3 Intelligent-Tiering zu übertragen. Sie können Daten direkt per PUT in S3 Intelligent-Tiering übertragen, indem Sie `INTELLIGENT_TIERING` im Header `x-amz-storage-class` angeben, oder S3-Lebenszykluskonfigurationen konfigurieren, um Objekte aus S3 Standard oder S3 Standard-Infrequent Access zu S3 Intelligent-Tiering zu übertragen.

Hochladen von Daten in S3 Intelligent-Tiering mit Direct PUT

Wenn Sie ein Objekt in die Speicherklasse S3 Intelligent-Tiering mit dem API-Vorgang PUT hochladen, geben Sie S3 Intelligent-Tiering im `x-amz-storage-class`-Anforderungs-Header an.

Die folgende Anforderung speichert das Image, `my-image.jpg`, im `myBucket`-Bucket. Die Anforderung verwendet den `x-amz-storage-class`-Header, um anzufordern, dass das Objekt mit der Speicherklasse S3 Intelligent-Tiering gespeichert ist.

Example

```
PUT /my-image.jpg HTTP/1.1
Host: myBucket.s3.<Region>.amazonaws.com (http://amazonaws.com/)
Date: Wed, 1 Sep 2021 17:50:00 GMT
Authorization: authorization string
Content-Type: image/jpeg
Content-Length: 11434
Expect: 100-continue
x-amz-storage-class: INTELLIGENT_TIERING
```

Übergang von Daten zu S3 Intelligent-Tiering von S3 Standard oder S3 Standard-Infrequent Access mit S3-Lebenszyklus

Sie können Regeln zu Ihrer S3-Lebenszykluskonfiguration hinzufügen und damit Amazon S3 anweisen, Objekte von einer Speicherklasse in eine andere zu wechseln. Informationen zu unterstützten Übergängen und zugehörigen Einschränkungen finden Sie unter Migrieren von Objekten mit S3-Lebenszyklus.

Sie können S3-Lebenszykluskonfigurationen auf der Bucket- oder Präfixebene angeben. In dieser S3-Lebenszyklus-Konfigurationregel spezifiziert der Filter ein Schlüsselpräfix (`documents/`). Aus diesem Grund gilt die Regel für Objekte mit dem Schlüsselnamenpräfix `documents/`, wie beispielsweise `documents/doc1.txt` und `documents/doc2.txt`. Die Regel gibt eine `Transition`-Aktion an, die Amazon S3 anweist, Objekte 0 Tage nach der Erstellung in die Speicherklasse S3 Intelligent-Tiering zu übertragen. In diesem Fall können Objekte nach der Erstellung um Mitternacht UTC auf S3 Intelligent-Tiering umgestellt werden.

Example

```
<LifecycleConfiguration>
  <Rule>
    <ID>ExampleRule</ID>
    <Filter>
       <Prefix>documents/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
```

```
      <Days>0</Days>
      <StorageClass>INTELLIGENT_TIERING</StorageClass>
    </Transition>
 </Rule>
</LifecycleConfiguration>
```

Aktivieren der Stufen S3 Intelligent-Tiering Archive Access und Deep Archive Access

Um die niedrigsten Speicherkosten für Daten zu erzielen, auf die innerhalb von Minuten bis Stunden zugegriffen werden kann, können Sie eine oder beide Archivzugriffsebenen aktivieren, indem Sie mithilfe der AWS Management Console-, AWS CLI- oder Amazon-S3-API eine Konfiguration auf Bucket-, Präfix- oder Objekt-Tag-Ebene erstellen.

Verwenden der S3-Konsole

So aktivieren Sie die automatische S3 Intelligent-Tiering-Archivierung

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des von Ihnen erstellten Buckets aus.
3. Wählen Sie Properties (Eigenschaften).
4. Navigieren Sie zu den S3 Intelligent-Tiering-Archive-Konfigurationen und wählen Sie Erstellen einer Konfiguration aus.
5. Geben Sie im Abschnitt Archiveinstellungen einen beschreibenden Konfigurationsnamen für die Konfiguration des S3 Intelligent-Tiering-Archivs an.
6. Wählen Sie unter Auswählen eines Konfigurationsbereichs einen zu verwendenden Konfigurationsbereich aus. Optional können Sie den Konfigurationsbereich auf bestimmte Objekte innerhalb eines Buckets beschränken, indem Sie ein gemeinsames Präfix, ein Objekt-Tag oder eine Kombination dieser beiden verwenden.
   a. Um den Umfang der Konfiguration einzuschränken, wählen Sie Begrenzen des Umfangs dieser Konfiguration mithilfe eines oder mehrerer Filter aus.
   b. Um den Bereich der Konfiguration mit einem einzigen Präfix zu beschränken, geben Sie den Präfix unter Präfix ein.
   c. Um den Umfang der Konfiguration mithilfe von Objekt-Tags einzuschränken, wählen Sie Hinzufügen eines Tags, und geben Sie einen Wert für Schlüssel ein.
7. Wählen Sie unter Status Aktivieren aus.

8. Wählen Sie im Abschnitt Archiveinstellungen eine oder beide der Archivzugriffsebenen aus, die aktiviert werden sollen.
9. Wählen Sie Erstellen aus.

Verwendung von AWS CLI

Zur Verwaltung der S3-Intelligent-Tiering-Konfigurationen können Sie die folgenden AWS CLI-Befehle verwenden:

- `delete-bucket-intelligent-tiering-configuration`
- `get-bucket-intelligent-tiering-configuration`
- `list-bucket-intelligent-tiering-configurations`
- `put-bucket-intelligent-tiering-configuration`

Weitere Informationen zum Einrichten der AWS CLI finden Sie unter Entwickeln mit Amazon S3 über die AWS CLI.

Bei der Verwendung der AWS CLI können Sie die Konfiguration nicht als XML-Datei angeben. Sie müssen stattdessen das JSON-Format angeben. Nachfolgend finden Sie Beispiele für XML-S3-Intelligent-Tiering-Konfigurationen und entsprechenden JSON-Code, den Sie in einem AWS CLI-Befehl angeben können.

Im folgenden Beispiel wird eine S3-Intelligent-Tiering-Konfiguration in den angegebenen Bucket übertragen.

Example **`put-bucket-intelligent-tiering-configuration`**

JSON

```
{
  "Id": "string",
  "Filter": {
    "Prefix": "string",
    "Tag": {
      "Key": "string",
      "Value": "string"
    },
    "And": {
      "Prefix": "string",
```

```
      "Tags": [
        {
          "Key": "string",
          "Value": "string"
        }
        ...
      ]
    }
  },
  "Status": "Enabled"|"Disabled",
  "Tierings": [
    {
      "Days": integer,
      "AccessTier": "ARCHIVE_ACCESS"|"DEEP_ARCHIVE_ACCESS"
    }
    ...
  ]
}
```

XML

```
PUT /?intelligent-tiering&id=Id HTTP/1.1
Host: Bucket.s3.amazonaws.com
<?xml version="1.0" encoding="UTF-8"?>
<IntelligentTieringConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Id>string</Id>
   <Filter>
      <And>
         <Prefix>string</Prefix>
         <Tag>
            <Key>string</Key>
            <Value>string</Value>
         </Tag>
         ...
      </And>
      <Prefix>string</Prefix>
      <Tag>
         <Key>string</Key>
         <Value>string</Value>
      </Tag>
   </Filter>
   <Status>string</Status>
   <Tiering>
```

```
      <AccessTier>string</AccessTier>
      <Days>integer</Days>
   </Tiering>
   ...
</IntelligentTieringConfiguration>
```

Verwenden des PUT-API-Vorgangs

Sie können den `PutBucketIntelligentTieringConfiguration`-Vorgang für einen angegebenen Bucket und bis zu 1000 S3-Intelligent-Tiering-Konfigurationen pro Bucket nutzen. Mit einem gemeinsamen Präfix oder Objekt-Tag können Sie festlegen, welche Objekte innerhalb eines Buckets für die Archivzugriffsstufen berechtigt sind. Durch die Verwendung eines gemeinsamen Präfixes oder Objekt-Tags können Sie auf bestimmte Geschäftsanwendungen, Workflows oder interne Organisationen ausrichten. Außerdem haben Sie die Flexibilität, die Stufe Archivzugriff, die Stufe Deep Archive Access oder beide zu aktivieren.

Erste Schritte mit S3 Intelligent-Tiering

Weitere Informationen zur Verwendung von S3 Intelligent-Tiering finden Sie unter Tutorial: Erste Schritte mit S3 Intelligent-Tiering.

Verwenden von S3 Intelligent-Tiering

Die Speicherklasse S3 Intelligent-Tiering ermöglicht automatische Speicherkosteneinsparungen in drei Zugriffsebenen mit niedriger Latenz und hohem Durchsatz. Sie bietet außerdem optionale Archivierungsfunktionen, mit denen Sie die niedrigsten Speicherkosten in der Cloud für Daten erzielen, auf die innerhalb von Minuten bis Stunden zugegriffen werden kann. Die Speicherklasse S3 Intelligent-Tiering unterstützt alle Amazon-S3-Funktionen, einschließlich der folgenden:

- S3 Inventory zur Überprüfung der Zugriffsebene von Objekten
- S3-Replikation, zum Replizieren von Daten in beliebiger AWS-Region
- S3 Storage Lens zum Anzeigen von Speichernutzungs- und Aktivitätsmetriken
- Serverseitige Verschlüsselung, für den Schutz von Objektdaten
- S3-Objektsperre, um versehentliches Löschen von Daten zu verhindern
- AWS PrivateLink, um über einen privaten Endpunkt in einer Virtual Private Cloud (VPC) auf Amazon S3 zuzugreifen

Festlegen welche der S3-Intelligent-Tiering-Zugriffsebene-Objekte gespeichert sind in

Wenn Sie eine Liste Ihrer Objekte und der zugehörigen Metadaten, einschließlich der Stufe des S3-Intelligent-Tiering-Zugriffs, erhalten möchten, können Sie Amazon S3 Inventory nutzen. S3 Inventory bietet CSV, ORC oder Parquet-Ausgabedateien, die Ihre Objekte und die entsprechenden Metadaten auflisten. Sie können diese Bestandsberichte entweder täglich oder wöchentlich für einen Amazon-S3-Bucket oder ein gemeinsames Präfix erhalten. (Gemeinsames Präfix bezieht sich auf Objekte, die Namen haben, die mit einer bestimmten Zeichenfolge beginnen.)

Anzeigen des Archivstatus eines Objekts in S3 Intelligent-Tiering

Wenn Sie eine Benachrichtigung erhalten möchten, sobald ein Objekt innerhalb der Speicherklasse S3 Intelligent-Tiering entweder auf die Stufe Archive Access oder die Stufe Deep Archive Access verschoben wurde, können Sie S3-Ereignisbenachrichtigungen einrichten. Weitere Informationen finden Sie unter Aktivieren von Ereignisbenachrichtigungen.

Amazon S3 kann Ereignisbenachrichtigungen in einem Amazon Simple Notification Service (Amazon SNS)-Thema, einer Amazon Simple Queue Service (Amazon SQS)-Warteschlange oder einer AWS Lambda-Funktion veröffentlichen. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Das Folgende ist ein Beispiel für eine Nachricht, die Amazon S3 sendet, um ein `s3:IntelligentTiering`-Ereignis zu veröffentlichen. Weitere Informationen finden Sie unter Struktur der Ereignisnachricht.

```
{
   "Records":[
      {
         "eventVersion":"2.3",
         "eventSource":"aws:s3",
         "awsRegion":"us-west-2",
         "eventTime":"1970-01-01T00:00:00.000Z",
         "eventName":"IntelligentTiering",
         "userIdentity":{
            "principalId":"s3.amazonaws.com"
         },
         "requestParameters":{
            "sourceIPAddress":"s3.amazonaws.com"
         },
         "responseElements":{
            "x-amz-request-id":"C3D13FE58DE4C810",
```

```
            "x-amz-id-2":"FMyUVURIY8/IgAtTv8xRjskZQpcIZ9KG4V5Wp6S7S/
JRWeUWerMUE5JgHvANOjpD"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"testConfigRule",
            "bucket":{
               "name":"mybucket",
               "ownerIdentity":{
                  "principalId":"A3NL1KOZZKExample"
               },
               "arn":"arn:aws:s3:::mybucket"
            },
            "object":{
               "key":"HappyFace.jpg",
               "size":1024,
               "eTag":"d41d8cd98f00b204e9800998ecf8427e",
            }
         },
         "intelligentTieringEventData":{
            "destinationAccessTier": "ARCHIVE_ACCESS"
            }
      }
   ]
}
```

Sie können auch eine HEAD-Objekt-Anfrage nutzen, um den Archivstatus eines Objekts anzuzeigen. Wenn ein Objekt in der Speicherklasse S3 Intelligent-Tiering gespeichert wird und sich in einer der Archivebenen befindet, zeigt die HEAD-Objektantwort die aktuelle Archivebene an. Um die Archivstufe anzuzeigen, verwendet die Anfrage den x-amz-archive-status-Header an.

Die folgende HEAD-Objektanforderung gibt die Metadaten eines Objekts zurück (in diesem Fall *my-image.jpg*).

Example

```
HEAD /my-image.jpg HTTP/1.1
Host: bucket.s3.region.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:02236Q3V0RonhpaBX5sCYVf1bNRuU=
```

Darüber hinaus können Sie HEAD-Objektanforderungen verwenden, um den Status einer `restore-object`-Anforderung zu überwachen. Wenn die Archivwiederherstellung ausgeführt wird, enthält die HEAD-Objekt-Antwort den x-amz-restore-Header.

Das folgende Beispiel veranschaulicht die HEAD-Objektantwort, die ein mit S3 Intelligent-Tiering archiviertes Objekt mit einer Wiederherstellungsanforderung anzeigt.

Example

```
HTTP/1.1 200 OK
x-amz-id-2: FSVaTMjrmBp3Izs1NnwBZeu7M19iI8UbxMbi0A8AirHANJBo+hEftBuiESACOMJp
x-amz-request-id: E5CEFCB143EB505A
Date: Fri, 13 Nov 2020 00:28:38 GMT
Last-Modified: Mon, 15 Oct 2012 21:58:07 GMT
ETag: "1accb31fcf202eba0c0f41fa2f09b4d7"
x-amz-storage-class: 'INTELLIGENT_TIERING'
x-amz-archive-status: 'ARCHIVE_ACCESS'
x-amz-restore: 'ongoing-request="true"'
x-amz-restore-request-date: 'Fri, 13 Nov 2020 00:20:00 GMT'
Accept-Ranges: bytes
Content-Type: binary/octet-stream
Content-Length: 300
Server: AmazonS3
```

Wiederherstellen von Objekten aus den Stufen S3 Intelligent-Tiering Archive Access und Deep Archive Access

Für Zugriff auf Objekte in den S3 Intelligent-Tiering Archive Access und Deep Archive Access müssen Sie eine Wiederherstellungsanforderung initiieren und warten, bis das Objekt in die Stufe für häufigen Zugriff verschoben wird, um darauf zuzugreifen. Weitere Informationen zu archivierten Objekten finden Sie unter Arbeiten mit archivierten Objekten.

Wenn Sie ein Objekt aus den Stufen Archive Access oder Deep Archive Access wiederherstellen, wird das Objekt zurück in die Stufe für häufige Zugriffe übergehen. Wenn anschließend nach 30 aufeinanderfolgenden Tagen nicht auf das Objekt zugegriffen wird, wird es automatisch in die Stufe „Infrequent Access“ verschoben. Das Objekt wechselt dann nach mindestens 90 aufeinanderfolgenden Tagen ohne Zugriff automatisch in die Stufe „Archive Access“. Das Objekt wechselt nach mindestens 180 aufeinanderfolgenden Tagen ohne Zugriff automatisch in die Stufe „Deep Archive Access“. Weitere Informationen finden Sie unter the section called “So funktioniert S3 Intelligent-Tiering”.

> **Note**
>
> Wenn Sie ein Objekt aus S3 Intelligent-Tiering wiederherstellen, fallen keine Abrufgebühren für Standard- oder Bulk-Abrufe an. Nachfolgende Wiederherstellungsanforderungen, die für archivierte Objekte aufgerufen werden, die bereits wiederhergestellt werden, werden jedoch als GET-Anforderung in Rechnung gestellt. Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

In der folgenden Tabelle werden die Abrufgeschwindigkeiten für archivierte Objekte zusammengefasst.

> **Note**
>
> Beschleunigte Abrufe sind eine Premium-Funktion, die für die S3-Intelligent-Tiering-Archivzugriffsebene verfügbar ist und mit der beschleunigten Anfrage und Abrufrate berechnet wird.
> Informationen zu Amazon-S3-Preisen finden Sie unter Amazon S3-Preise.

Sie können ein archiviertes Objekt mithilfe der Amazon-S3-Konsole, S3 Batch Operations, der REST-API und der AWS Command Line Interface (AWS CLI) wiederherstellen.

Verwenden der S3-Konsole

Wiederherstellung eines Objekts mit der Amazon-S3-Konsole

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der die Objekte enthält, die Sie wiederherstellen möchten.
3. Aktivieren Sie in der Liste Objekte das Kontrollkästchen neben mindestens einem der Objekte, die Sie wiederherstellen möchten. Wählen Sie Aktionen und dann Wiederherstellung aus S3 Intelligent-Tiering Archive Access oder Deep Archive Access aus.
4. Wählen Sie Restore (Wiederherstellen) aus.

> **Note**
>
> Objekte aus den Stufen S3 Intelligent-Tiering Archive Access und Deep Archive Access werden automatisch in der Stufe für häufige Zugriffe wiederhergestellt.

Verwendung von AWS CLI

Wenn Sie Objekte aus den Stufen S3 Intelligent-Tiering Archive Access und Deep Archive Access wiederherstellen möchten, verwenden Sie den `restore-object`-Befehl.

Der folgende Beispielbefehl stellt das Objekt *`dir1/example.obj`* im Bucket *`DOC-EXAMPLE-BUCKET`* wieder her. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3api restore-object --bucket DOC-EXAMPLE-BUCKET --key dir1/example.obj --restore-request '{}'
```

Sie können den folgenden beispielhaften Befehl verwenden, um den Status Ihrer `restore-object`-Anforderung zu überwachen. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3api head-object --bucket DOC-EXAMPLE-BUCKET --key dir1/example.obj
```

Weitere Informationen finden Sie unter restore-object in der Referenz zum AWS CLI-Befehl.

> **Note**
>
> Im Gegensatz zu den Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive akzeptieren Wiederherstellungsanforderungen für S3-Intelligent-Tiering-Objekte den `Days`-Wert nicht.

Verwenden der REST-API

Amazon S3 stellt einen API-Vorgang für Sie bereit, um eine Wiederherstellung des Archivs zu initiieren. Weitere Informationen finden Sie unter RestoreObject in der API-Referenz zu Amazon Simple Storage Service.

Verwendung von S3 Batch Operations

Zur Wiederherstellung von mehr als einem archivierten Objekt in einer einzigen Anforderung können Sie S3 Batch Operations nutzen. Sie stellen S3 Batch Operations eine Liste von Objekten zur Verfügung, für die Vorgänge ausgeführt werden sollen. S3-Batchoperationen rufen die entsprechende API-Operation auf, um die angegebene Operation auszuführen. Ein einzelner Batch-Vorgangsauftrag kann die angegebene Operation für Milliarden von Objekten ausführen, die Exabytes von Daten enthalten.

Um einen Batch-Operations-Auftrag zu erstellen, benötigen Sie ein Manifest, das nur die Objekte enthält, die Sie wiederherstellen möchten. Sie können mithilfe von S3 Inventory ein Manifest erstellen oder eine CSV-Datei mit den erforderlichen Informationen bereitstellen. Weitere Informationen finden Sie unter the section called “Angeben eines Manifests”.

Bevor Sie S3-Batch-Operations-Aufträge erstellen und ausführen, müssen Sie Amazon S3 die Erlaubnis erteilen, S3-Batch-Operations in Ihrem Namen durchzuführen. Die erforderlichen Berechtigungen finden Sie unter the section called “Gewähren von Berechtigungen”.

> **Note**
>
> Batch-Operations-Aufträge können entweder mit Objekte der Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive oder mit Objekten der Speicherstufen S3 Intelligent-Tiering Archive Access und Deep Archive Access arbeiten. Batch Operations kann im selben Auftrag nicht für beide Typen von archivierten Objekten ausgeführt werden. Um Objekte beider Typen wiederherzustellen, müssen Sie separate Batchoperations-Aufgaben erstellen.
>
> Weitere Informationen zur Verwendung von Batch Operations zum Wiederherstellen archivierter Objekte finden Sie unter the section called “Wiederherstellen von Objekten”.

So erstellen Sie einen S3-Auftrag „Initiate Restore Object Batch Operations“

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie im linken Navigationsbereich auf Batchvorgänge.
3. Wählen Sie Create job (Auftrag erstellen) aus.
4. Wählen Sie für AWS-Region die Region aus, in der Sie Ihren Auftrag erstellen möchten.

5. Wählen Sie unter Manifestformat das zu verwendende Manifest aus.

   - Wenn Sie S3-Bestandsbericht auswählen, geben Sie den Pfad zum `manifest.json`-Objekt ein, das Amazon S3 als Teil des Bestandsberichts im CSV-Format generiert hat. Wenn Sie nicht das aktuelle Manifest verwenden möchten, können Sie optional die Versions-ID des `manifest.json`-Objekts angeben.
   - Wenn Sie CSV auswählen, geben Sie den Pfad zu einem CSV-formatierten Manifestobjekt ein. Das Manifestobjekt muss das in der Konsole beschriebene Format befolgen. Wenn Sie nicht die aktuelle Version verwenden möchten, können Sie optional die Versions-ID des Manifestobjekts angeben.

6. Wählen Sie Next (Weiter).
7. Wählen Sie im Abschnitt Operation die Option Wiederherstellen aus.
8. Wählen Sie im Abschnitt Wiederherstellen bei Quelle wiederherstellen die Stufe Intelligent-Tiering Archive Access oder Deep Archive Access aus. Wählen Sie bei Abrufstufe die zu verwendende Stufe aus.
9. Wählen Sie Next (Weiter).
10. Füllen Sie auf der Seite Zusätzliche Optionen konfigurieren die folgenden Abschnitte aus:

    - Geben Sie im Abschnitt Zusätzliche Optionen eine Beschreibung des Auftrags und eine Prioritätsnummer für den Auftrag an. Höhere Nummern bedeuten eine höhere Priorität. Weitere Informationen finden Sie unter the section called “Zuweisen der Auftragspriorität”.
    - Wählen Sie im Abschnitt Abschlussbericht aus, ob Batch Operations einen Abschlussbericht erstellen soll. Weitere Informationen zu den Fertigstellungsberichten finden Sie unter the section called “Abschlussberichte”.
    - Im Abschnitt Berechtigungen müssen Sie Amazon S3 die Erlaubnis erteilen, Batch Operations in Ihrem Namen durchzuführen. Die erforderlichen Berechtigungen finden Sie unter the section called “Gewähren von Berechtigungen”.
    - (Optional) fügen Sie im Abschnitt Auftrags-Tags Tags in Schlüssel-Wert-Paaren hinzu. Weitere Informationen finden Sie unter the section called “Verwenden von Markierungen”.

    Wählen Sie Weiter aus, sobald Sie fertig sind.

11. Überprüfen Sie die Einstellungen auf der Seite Review. Wenn Sie Änderungen vornehmen müssen, wählen Sie Previous. Wählen Sie andernfalls Auftrag erstellen.

Weitere Informationen über Batch Operations finden Sie unter Objekte mit Batch Operations wiederherstellen und Erstellen eines S3-Batch-Vorgangsauftrags.

Überprüfen des Wiederherstellungsstatus eines Objekts

Sie können den Fortschritt der Wiederherstellung Ihres Objekts auf der Detailseite des Objekts in der Amazon-S3-Konsole, der AWS CLI oder der REST-API überprüfen. Weitere Informationen finden Sie unter Überprüfung des Wiederherstellungsstatus und des Ablaufdatums.

Sie können sich über den Abschluss der Objektwiederherstellung benachrichtigen lassen, indem Sie die Aktion `s3:ObjectRestore:Completed` mit der Funktion Amazon-S3-Ereignisbenachrichtigungen verwenden.

Verwalten Ihres Speicher-Lebenszyklus

Um Ihre Objekte so zu verwalten, dass diese während ihres gesamten Lebenszyklus kosteneffizient gespeichert werden, konfigurieren Sie deren Amazon-S3-Lebenszyklus. Eine S3-Lebenszyklus-Konfiguration besteht aus einer Reihe von Regeln, mit denen Aktionen definiert werden, die Amazon S3 auf eine Gruppe von Objekten anwendet. Es gibt zwei Aktionstypen:

- Übergangsaktionen – Diese Aktionen definieren, wann Objekte in eine andere Speicherklasse übergehen. Beispielsweise können Sie festlegen, dass Objekte 30 Tage nach ihrer Erstellung in die Speicherklasse S3 Standard-IA (IA steht für „Infrequent Access“, seltener Zugriff) übergehen und ein Jahr nach ihrer Erstellung in der Speicherklasse S3 Glacier Flexible Retrieval archiviert werden sollen. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.

  Mit Lebenszyklus-Überführungsanforderungen sind Kosten verbunden. Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.
- Ablaufaktionen – Diese Aktionen definieren, wann Objekte ablaufen. In Amazon S3 werden die abgelaufenen Objekte für Sie gelöscht.

  Die Lebenszyklusablaufkosten sind von dem Zeitpunkt abhängig, an dem Objekte ablaufen sollen. Weitere Informationen finden Sie unter Auslaufende Objekte.

Wenn es zwischen dem Zeitpunkt, an dem ein Objekt für eine Lebenszyklusaktion berechtigt wird, und dem Zeitpunkt, zu dem Amazon S3 Ihr Objekt überträgt oder abläuft, zu Verzögerungen kommt, werden die Abrechnungsänderungen angewendet, sobald das Objekt für die Lebenszyklusaktion berechtigt ist. Wenn beispielsweise der Ablauf eines Objekts geplant ist und Amazon S3 das Objekt

nicht sofort abläuft, wird Ihnen nach Ablauf der Ablaufzeit keine Speichergebühr berechnet. Die einzige Ausnahme von diesem Verhalten ist, wenn Sie eine Lebenszyklusregel für den Übergang zur S3-Intelligent-Tiering-Speicherklasse haben. In diesem Fall treten Abrechnungsänderungen erst auf, wenn das Objekt auf S3 Intelligent-Tiering übergegangen ist.

Weitere Informationen zu S3-Lebenszyklusregeln finden Sie unter Elemente der Lebenszyklus-Konfiguration.

Wenn Sie detaillierte Metriken für S3 Lifecycle abrufen möchten, können Sie Metriken von Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. S3 Storage Lens bietet Metriken zur Anzahl der S3-Lifecycle-Regeln sowie Metriken, anhand derer Sie Buckets identifizieren können, für die S3 Versioning aktiviert ist oder ein hoher Prozentsatz an Bytes nicht aktueller Versionen vorliegt. Weitere Informationen finden Sie unter Verwenden von S3 Storage Lens zur Optimierung Ihrer Speicherkosten.

Verwalten des Objektlebenszyklus

Für Objekte mit vorgegebenem Lebenszyklus können Sie S3-Lebenszyklus-Konfigurationsregeln definieren. z. B.:

- Wenn Sie regelmäßig Protokolle in einen Bucket hochladen, werden diese möglicherweise nur für eine Woche oder einen Monat von Ihrer Anwendung benötigt. Anschließend sollen diese gelöscht werden.
- Auf einige Dokumente erfolgt für einen begrenzten Zeitraum ein häufiger Zugriff. Danach wird nur selten auf diese zugegriffen. Irgendwann benötigen Sie keinen Echtzeitzugriff mehr auf die Dokumente, aber seitens Ihrer Organisation oder der Gesetzgebung müssen Sie diese noch für einen gewissen Zeitraum archivieren. Nach Ablauf dieser Zeitspanne können Sie sie löschen.
- Sie können auch einige Datentypen primär für Archivierungszwecke in Amazon S3 hochladen. Beispielsweise können Sie digitale Medienarchive, Datensätze des Finanz- und Gesundheitswesens, rohe Genomsequenzdaten, langfristige Datenbanksicherungen oder Daten, die zur Einhaltung gesetzlicher Vorschriften aufbewahrt werden müssen, archivieren.

Mithilfe den S3-Konfigurationsregeln für den Lebenszyklus können Sie Amazon S3 anweisen, Objekte in kostengünstigere Speicherklassen zu übergeben bzw. zu archivieren oder zu löschen.

Erstellen einer Lebenszyklus-Konfiguration

Eine S3-Lebenszyklus-Konfiguration ist eine XML-Datei, die aus einem Satz von Regeln mit vordefinierten Aktionen besteht, die Amazon S3 für Objekte während ihrer Lebensdauer ausführen soll.

Sie können den Lebenszyklus auch mithilfe der Amazon-S3-Konsole, der REST-API, der AWS-SDKs und der AWS Command Line Interface (AWS CLI) konfigurieren. Weitere Informationen finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

Amazon S3 unterstützt verschiedene REST-API-Vorgänge, mit denen Sie die Lebenszyklus-Konfiguration für einen Bucket verwalten können. Amazon S3 speichert die Konfiguration als Lebenszyklus-Subressource, die Ihrem Bucket zugeordnet ist. Details dazu finden Sie unter:

PUT Bucket-Lebenszyklus

GET Bucket-Lebenszyklus

DELETE Bucket-Lebenszyklus

Weitere Informationen zum Erstellen einer Lebenszyklus-Konfiguration finden Sie in den folgenden Themen:

Themen

- Übergang von Objekten mit Amazon-S3-Lebenszyklus
- Auslaufende Objekte
- Festlegen der Lebenszyklus-Konfiguration für einen Bucket
- Lebenszyklus- und andere Bucket-Konfigurationen
- Konfigurieren von Lebenszyklus-Ereignisbenachrichtigungen
- Elemente der Lebenszyklus-Konfiguration
- Beispiele der S3-Lebenszyklus-Konfiguration

Übergang von Objekten mit Amazon-S3-Lebenszyklus

Sie können Regeln zu Ihrer S3-Lebenszykluskonfiguration hinzufügen und damit Amazon S3 anweisen, Objekte in eine andere Amazon-S3-Speicherklasse zu übergeben. Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen. Einige Beispiele

dafür, wann Sie S3-Lebenszykluskonfigurationen auf diese Weise verwenden könnten, sind die folgenden:

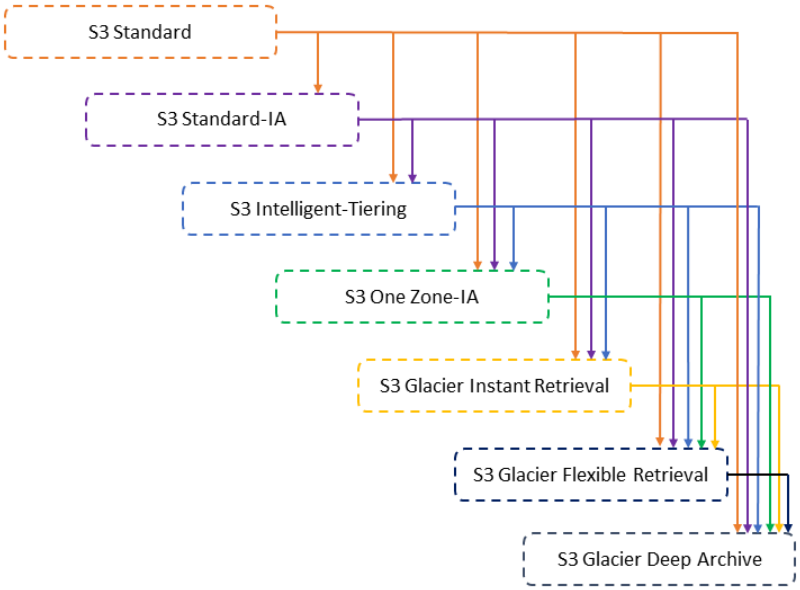
- Wenn Sie wissen, dass nur selten auf bestimmte Objekte zugegriffen wird, können Sie diese in die Speicherklasse S3 Standard-IA übergeben.
- Sie können Objekte, für die kein Echtzeitzugriff mehr benötigt wird, in der Speicherklasse S3 Glacier Flexible Retrieval archivieren.

Die folgenden Abschnitte beschreiben unterstützte Übergange, zugehörige Einschränkungen und den Übergang in die Speicherklasse S3 Glacier Flexible Retrieval.

Unterstützte Transaktionen und zugehörige Einschränkungen

Sie können in einer S3-Lebenszyklus-Konfiguration Regeln definieren, um Objekte aus einer Speicherklasse zu einer anderen Speicherklasse zu überführen, um Speicherkosten zu sparen. Wenn Sie die Zugriffsmuster Ihrer Objekte nicht kennen oder sich Ihre Zugriffsmuster über die Zeit verändern, können Sie die Objekte zur Speicherklasse S3 Intelligent-Tiering überführen, um automatische Kosteneinsparungen zu erzielen. Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.

Amazon S3 unterstützt das Wasserfallmodell für die Überführung zwischen Speicherklassen wie im folgenden Diagramm gezeigt.



Unterstützte Lebenszyklusübergänge

Amazon S3 unterstützt die folgenden Lebenszyklusübergänge zwischen Speicherklassen mittels einer S3-Lebenszykluskonfiguration.

Die folgenden Übergänge sind möglich:

- Von der Speicherklasse S3 Standard zu einer beliebigen anderen Speicherklasse.
- Von der Speicherklasse S3-Standard-IA zu den Speicherklassen S3 Intelligent-Tiering, S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive.
- Von der Speicherklasse S3 Intelligent-Tiering zu den Speicherklassen S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive.
- Von der Speicherklasse S3 One Zone-IA zu den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive.
- Von der Speicherklasse S3 Glacier Instant Retrieval zu den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive.
- Die Speicherklasse S3 Glacier Flexible Retrieval zur Speicherklasse S3 Glacier Deep Archive.
- Von einer beliebigen Speicherklasse zur Speicherklasse S3 Glacier Deep Archive.

> **Note**
>
> Für Lebenszyklusübergänge fallen keine Gebühren für den Datenabruf an. Es fallen jedoch Gebühren pro Anforderung an, wenn Sie PUT-, COPY- oder Lebenszyklusregeln verwenden, um Daten in eine beliebige S3-Speicherklasse zu verschieben. Berücksichtigen Sie die Kosten für die Aufnahme oder Übertragung, bevor Sie Objekte in eine beliebige Speicherklasse verschieben. Weitere Information zu Kostenaspekten finden Sie unter Amazon S3 – Preise.

Nicht unterstützte Lebenszyklusübergänge

Amazon S3 unterstützt keine der folgenden Lebenszyklusübergänge.

Die folgenden Übergänge sind nicht möglich:

- Von jeder Speicherklasse zur Speicherklasse S3 Standard.
- Von jeder Speicherklasse zur Reduced-Redundancy-Speicherklasse (RRS).
- Von der Speicherklasse S3 Intelligent-Tiering zur Speicherklasse S3 Standard-IA
- Von der S3-One-Zone-IA-Speicherklasse zu den Speicherklassen S3 Intelligent-Tiering, S3 Standard-IA und S3 Glacier Instant Retrieval.

Beschränkungen

Für die Übergänge der Lebenszyklus-Speicherklassen gelten folgende Einschränkungen:

Objektgröße und Übergänge von S3 Standard oder S3 Standard-IA zu S3 Intelligent-Tiering, S3 Standard-IA oder S3 One Zone-IA

Beim Übergang von Objekten von den Speicherklassen S3 Standard oder S3 Standard-IA zu S3 Intelligent-Tiering, S3 Standard-IA oder S3 One Zone-IA gelten die folgenden Objektgrößenbeschränkungen:

- Größere Objekte – Für die folgenden Übergänge gibt es einen Kostenvorteil beim Übergang größerer Objekte:
  - Von den Speicherklassen S3 Standard oder S3 Standard-IA zu S3 Intelligent-Tiering.
  - Von der Speicherklasse S3 Standard zu S3 Standard-IA oder S3 One Zone-IA.

- Objekte kleiner als 128 KiB: Für die folgenden Übergänge wird für Objekte, die kleiner als 128 KiB sind, kein Übergang in Amazon S3 durchgeführt:
  - Von den Speicherklassen S3 Standard oder S3 Standard-IA zu S3 Intelligent-Tiering oder S3 Glacier Instant Retrieval.
  - Von der Speicherklasse S3 Standard zu S3 Standard-IA oder S3 One Zone-IA.

> **Note**
>
> Sie können Lebenszyklusregeln basierend auf der Objektgröße filtern.

> **Important**
>
> Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere S3-Lebenszyklus-Aktionen auszuführen sind. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:
>
> - Das permanente Löschen hat Vorrang vor einem Übergang.
> - Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
> - Wenn ein Objekt sowohl für einen S3 Glacier Flexible Retrieval als auch für einen S3 Standard-IA (oder S3 One Zone-IA) -Übergang in Frage kommt, entscheidet sich Amazon S3 für den Übergang zu S3 Glacier Flexible Retrieval.
>
> Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht.

Mindestanzahl der Tage für den Übergang in S3 Standard-IA oder S3 One Zone-IA

Bevor Sie Objekte in S3 Standard-IA oder S3 One Zone-IA überführen, müssen diese mindestens 30 Tage in Amazon S3 gespeichert worden sein. Beispielsweise können Sie keine Lebenszyklusregel erstellen, mit der Objekte einen Tag nach dem Erstellungsdatum in die Speicherklasse S3 Standard-IA übergehen. Amazon S3 unterstützt diesen Übergang erst nach 30 Tagen, da auf neuere Objekte meist häufiger zugegriffen wird oder sie schneller gelöscht werden, als das für S3 Standard-IA- oder S3 One Zone-IA-Speicher sinnvoll wäre.

Beim Übergang von nicht aktuellen Objekten (in versionsfähigen Buckets) können dementsprechend nur Objekte, die seit mindestens 30 Tagen nicht mehr aktuell sind, in S3 Standard-IA- oder S3 One Zone-IA-Speicher übergehen. Eine Liste der Mindestspeicherdauer für alle Speicherklassen finden Sie unter Vergleich der Amazon-S3-Speicherklassen.

Minimale 30-tägige Speichergebühr für , S3-Standard-IA und S3 One Zone-IA

Die Speicherklassen S3 Standard-IA und S3 One Zone-IA haben eine Mindestspeichergebühr von 30 Tagen. Daher können Sie keine einzige Lebenszyklusregel für Übergänge zu S3 Standard-IA oder S3 One Zone-IA und S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive angeben, wenn der Übergang zu S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive früher als 30 Tage nach dem Übergang zu S3 Standard-IA oder S3 One Zone-IA auftritt.

Derselbe Mindestzeitraum von 30 Tagen gilt auch für den Übergang von S3 Standard-IA-Speicher in S3 One Zone-IA. Sie können dafür zwei Regeln spezifizieren, jedoch fallen die Mindestgebühren für Speicher an. Weitere Information zu Kostenaspekten finden Sie unter Amazon S3 – Preise.

Verwalten des vollständigen Lebenszyklus eines Objekts

Sie können diese S3-Lebenszyklus-Aktionen kombinieren, um den vollständigen Lebenszyklus eines Objekts zu verwalten. Angenommen, die von Ihnen erstellten Objekte haben einen definierten Lebenszyklus. Anfänglich erfolgt ein häufiger Zugriff auf die Objekte für einen Zeitraum von 30 Tagen. Danach erfolgt nur noch ein seltener Zugriff für bis zu 90 Tage. Anschließend werden die Objekte nicht mehr benötigt, daher können Sie entscheiden, sie zu archivieren oder zu löschen.

In diesem Szenario erstellen Sie eine S3-Lebenszyklusregel, in der Sie eine erste Überführungsaktion zu S3 Intelligent-Tiering, S3 Standard-IA oder S3 One Zone-IA, eine weitere Überführungsaktion zu S3 Glacier Flexible Retrieval zu Archivierungszwecken und schließlich eine Ablaufaktion angeben. Beim Übergang der Objekte von einer Speicherklasse in eine andere können Sie Kosten für Speicher sparen. Weitere Information zu Kostenaspekten finden Sie unter Amazon S3 – Preise.

Übergang in die Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive (Objektarchivierung)

Mithilfe der S3-Lebenszyklus-Konfiguration können Sie Objekte zur Archivierung in die Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive überführen. Wenn Sie die Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive wählen, bleibt Ihr Objekt in Amazon S3. Sie können nicht direkt über den separaten Amazon-S3-Glacier-Service auf sie

zugreifen. Allgemeine Informationen zu S3 Glacier finden Sie unter Was ist S3 Glacier im Amazon-S3-Glacier-Entwicklerhandbuch.

Bevor Sie Ihre Objekte archivieren, lesen Sie die folgenden Abschnitte, wo Sie weitere relevante Aspekte finden.

Allgemeine Überlegungen

Bevor Sie Objekte archivieren, sollten Sie die folgenden allgemeinen Überlegungen in Betracht ziehen:

- Verschlüsselte Objekte bleiben während des gesamten Übergangsprozesses der Speicherklasse verschlüsselt.
- Objekte, die in den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeichert sind, sind nicht in Echtzeit verfügbar.

  Archivierte Objekte sind Amazon-S3-Objekte, aber um auf ein archiviertes Objekt zugreifen zu können, müssen Sie zuerst eine temporäre Kopie davon wiederherstellen. Die wiederhergestellte Objektkopie steht nur für die Dauer zur Verfügung, die Sie in der Wiederherstellungsanforderung angegeben haben. Danach löscht Amazon S3 die temporäre Kopie und das Objekt bleibt in S3 Glacier Flexible Retrieval archiviert.

  Sie können ein Objekt mit Hilfe der Amazon-S3-Konsole wiederherstellen, oder programmgesteuert mit Hilfe der AWS-SDK-Wrapper-Bibliotheken oder über die Amazon S3 REST-API in Ihrem Code. Weitere Informationen finden Sie unter Wiederherstellen eines archivierten Objekts.
- Objekte, die in der Speicherklasse S3 Glacier Flexible Retrieval gespeichert sind, können nur in die Speicherklasse S3 Glacier Deep Archive überführt werden.

  Sie können mit einer S3-Lebenszyklusregel die Speicherklasse eines Objekts von S3 Glacier Flexible Retrieval nur zur Speicherklasse S3 Glacier Deep Archive ändern. Wenn Sie die Speicherklasse eines in S3 Glacier Flexible Retrieval gespeicherten Objekts zu einer anderen Speicherklasse als S3 Glacier Deep Archive ändern möchten, müssen Sie die Wiederherstellungsoperation verwenden, um zunächst eine temporäre Kopie des Objekts zu erstellen. Anschließend verwenden Sie die Kopieroperation, um das Objekt zu überschreiben und dabei S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 One Zone-IA oder Reduced Redundancy als Speicherklasse anzugeben.
- Die Überführung von Objekten in die Speicherklasse S3 Glacier Deep Archive ist nicht umkehrbar.

Sie können nicht mit einer S3-Lebenszyklusregel die Speicherklasse eines Objekts von S3 Glacier Deep Archive zu einer anderen Speicherklasse ändern. Wenn Sie die Speicherklasse eines archivierten Objekts in eine andere Speicherklasse ändern möchten, müssen Sie die Wiederherstellungsoperation verwenden, um zunächst eine temporäre Kopie des Objekts zu erstellen. Verwenden Sie dann den Kopiervorgang, um das Objekt zu überschreiben, indem Sie S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval oder Reduced Redundancy Storage als Speicherklasse angeben.

> **Note**
>
> Die Copy-Operation für wiederhergestellte Objekte wird in der Amazon-S3-Konsole für Objekte in der Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive nicht unterstützt. Verwenden Sie für diese Art von Kopiervorgang die AWS Command Line Interface (AWS CLI), die AWS-SDKs oder die REST-API.

Die Objekte, die in den Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive gespeichert sind, sind nur über Amazon S3 sichtbar und verfügbar. Sie sind nicht über den separaten Amazon-S3-Glacier-Service verfügbar.

Es handelt sich jedoch um Amazon-S3-Objekte, und Sie können nur über die Amazon-S3-Konsole oder die Amazon-S3-API darauf zugreifen. Sie können auf die archivierten Objekte nicht über die separate Amazon-S3-Konsole oder die Amazon-S3-Glacier-API zugreifen.

Kostenüberlegungen

Wenn Sie vorhaben, Daten mit seltenem Zugriff für einen Zeitraum von Monaten oder Jahren zu archivieren, können die Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive Ihre Speicherkosten reduzieren. Sie sollten jedoch Folgendes berücksichtigen, um sicherzustellen, dass die Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive geeignet für Sie ist:

- Gebühren für zusätzlichen Speicheraufwand – Wenn Sie Objekte zur Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive überführen, wird jedem Objekt eine feste Speichermenge hinzugefügt, um die Metadaten für die Verwaltung des Objekts zu berücksichtigen.
  - Für jedes in S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive archivierte Objekt verwendet Amazon S3 8 KB Speicher für den Namen des Objekts und andere Metadaten.

Amazon S3 speichert diese Metadaten, sodass Sie in Echtzeit eine Auflistung Ihrer archivierten Objekte mit der Amazon S3 API erhalten können. Weitere Informationen finden Sie unter Get Bucket (List Objects). Für diesen zusätzlichen Speicherplatz werden Ihnen -S3-Standardgebühren in Rechnung gestellt.

- Für jedes Objekt, das in S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive archiviert wird, fügt Amazon S3 32 KB Speicher für Index und zugehörige Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederherzustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen die Gebühren für S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive in Rechnung gestellt.

Wenn Sie kleine Objekte archivieren, sollten Sie diese Speichergebühren berücksichtigen. Ziehen Sie auch in Betracht, viele kleine Objekte in wenigen großen Objekten zusammenzufassen, um Kosten für den Verwaltungsaufwand zu reduzieren.

- Anzahl der Tage, für die die Objekte archiviert werden sollen – S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive sind langfristige Archivierungslösungen. Die minimale Speicherdauer beträgt 90 Tage für die Speicherklasse S3 Glacier Flexible Retrieval und 180 Tage für S3 Glacier Deep Archive. Beim Löschen von Daten, die in Amazon S3 Glacier archiviert wurden, fallen keine Gebühren an, wenn die von Ihnen gelöschten Objekte länger als die minimale Speicherdauer archiviert werden. Wenn Sie ein archiviertes Objekt innerhalb der minimalen Speicherdauer löschen oder überschreiben, stellt Amazon S3 eine anteilige Gebühr für das vorzeitige Löschen in Rechnung. Weitere Informationen zu den Kosten für die vorzeitige Löschungen finden Sie unter der Frage „Wie wird das Löschen von Objekten aus Amazon S3 Glacier berechnet, die weniger als 90 Tage alt sind?“ unter häufig gestellte Fragen zu Amazon S3.
- Gebühren für Übergangsanfragen von S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive – Jedes Objekt, das Sie auf die Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive umstellen, stellt eine Übergangsanfrage dar. Für jede dieser Anforderungen entstehen Kosten. Wenn Sie vorhaben, sehr viele Objekte zu überführen, sollten Sie die Anforderungskosten in Betracht ziehen. Wenn Sie eine Mischung von Objekten archivieren, die kleine Objekte enthält, insbesondere Objekte unter 128KB, empfehlen wir, den Filter für die Lebenszyklusobjektgröße zu verwenden, um kleine Objekte aus Ihrem Übergang herauszufiltern, um die Anforderungskosten zu senken. S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive blockieren den Übergang von Objekten unter 128KB nicht automatisch.
- Datenwiederherstellungsgebühren für S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive – S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive wurden für die Langzeitarchivierung von Daten entwickelt, auf die Sie selten zugreifen. Informationen zu Gebühren für die Datenwiederherstellung finden Sie unter der Frage „Wie viel kostet das Abrufen von Daten

aus Amazon S3 Glacier?“ unter häufig gestellte Fragen zu Amazon S3. Informationen zum Wiederherstellen von Daten aus Amazon S3 Glacier finden Sie unter Wiederherstellen eines archivierten Objekts.

Wenn Sie Objekte mithilfe der S3-Lebenszyklusverwaltung in Amazon S3 Glacier archivieren, führt Amazon S3 diese Objekte asynchron über. Es kann eine Verzögerung zwischen dem Übergangsdatum in der S3-Lebenszyklus-Konfigurationsregel und dem Datum des physischen Übergangs geben. Ihnen werden die Preise von Amazon S3 Glacier basierend auf dem in der Regel angegebenen Übergangsdatum berechnet. Weitere Informationen finden Sie im Abschnitt zu Amazon S3 Glacier in den Amazon S3 FAQ.

Die Amazon-S3-Produktdetailseite enthält Preisinformationen und Beispielberechnungen für die Archivierung von Amazon-S3-Objekten. Weitere Informationen finden Sie unter den folgenden Themen:

- Wie werden die Speicherkosten für in Amazon S3 Glacier archivierte Amazon-S3-Objekte berechnet? unter häufig gestellte Fragen zu Amazon S3.
- „Wie wird mir das Löschen von Objekten aus Amazon S3 Glacier in Rechnung gestellt, die weniger als 90 Tage alt sind?“ unter häufig gestellte Fragen zu Amazon S3.
- „Was kostet das Abrufen von Daten aus Amazon S3 Glacier?“ unter häufig gestellte Fragen zu Amazon S3.
- Amazon S3 – Preise für die Speicherkosten der unterschiedlichen Speicherklassen.

Wiederherstellen archivierter Objekte

Ein Zugriff auf archivierte Objekte ist nicht in Echtzeit möglich. Sie müssen zuerst eine Anforderung zur Wiederherstellung initiieren und dann warten, bis eine temporäre Kopie des Objekts für die Dauer bereitgestellt wird, die Sie in der Anforderung angegeben haben. Nachdem Sie eine temporäre Kopie des wiederhergestellten Objekts erhalten haben, bleibt die Speicherklasse des Objekts S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive. (Eine HEAD Object- oder GET Object-API-Operationsanforderung gibt S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive als Speicherklasse zurück.)

> **Note**
>
> Bei einer Wiederherstellung eines Archivs zahlen Sie sowohl für das Archiv (Tarif für S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive) als auch für die temporär wiederhergestellte Kopie (S3-Standard-Speichertarif). Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

Für den Zugriff auf ein kopiertes Objekt müssen Sie eine Anfrage zur Wiederherstellung initiieren. Das kann programmgesteuert oder über die Amazon-S3-Konsole erfolgen. Amazon S3 verarbeitet jeweils nur eine Anfrage pro Objekt. Weitere Informationen finden Sie unter Wiederherstellen eines archivierten Objekts.

Auslaufende Objekte

Wenn ein Objekt aufgrund seiner Lebenszykluskonfiguration das Ende seiner Lebensdauer erreicht hat, ergreift Amazon S3 eine Aktion basierend auf dem Status, in dem sich der Bucket befindet.

- Nicht versionierter Bucket – Amazon S3 stellt das Objekt zum Entfernen in eine Warteschlange und entfernt es asynchron und dauerhaft.
- Bucket mit aktiviertem Versioning – Wenn die aktuelle Objektversion keine Löschmarkierung ist, fügt Amazon S3 die Löschmarkierung mit einer eindeutigen Versions-ID hinzu. Damit ist die aktuelle Version nicht mehr aktuell und die Löschmarkierung wird zur aktuellen Version.
- Bucket mit ausgesetztem Versioning – Amazon S3 erstellt eine Löschmarkierung mit der Versions-ID null. Diese Löschmarkierung ersetzt jede Objektversion mi einer Versions-ID von null in der Versionshierarchie, womit das Objekt effektiv gelöscht wird.

Für einen versionsfähigen Bucket (d. h., das Versioning ist aktiviert oder ausgesetzt) gibt es mehrere Aspekte, die bestimmen, wie Amazon S3 die Ablaufaktion verarbeitet. Für Buckets mit aktivierter oder ausgesetzter Versionsverwaltung gilt Folgendes:

- Die Objektablaufaktion wird nur auf die aktuelle Version eines Objekts angewendet (sie wirkt sich nicht auf nicht aktuelle Objektversionen aus).
- Amazon S3 führt keine Aktion aus, wenn es eine oder mehrere Objektversionen gibt und die Löschmarkierung die aktuelle Version ist.
- Wenn die aktuelle Objektversion die einzige Objektversion und auch eine Löschmarkierung ist (auch als Löschmarkierung eines abgelaufenen Objekts bezeichnet, wobei alle Objektversionen

gelöscht werden und nur noch eine Löschmarkierung beibehalten wird), entfernt Amazon S3 die Löschmarkierung des abgelaufenen Objekts. Sie können die Ablaufaktion auch verwenden, um Amazon S3 anzuweisen, alle abgelaufenen Löschmarkierungen zu entfernen. Ein Beispiel finden Sie unter Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte.

Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Important

Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere S3-Lebenszyklus-Aktionen auszuführen sind. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:

- Das permanente Löschen hat Vorrang vor einem Übergang.
- Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
- Wenn ein Objekt sowohl für einen S3 Glacier Flexible Retrieval als auch für einen S3 Standard-IA (oder S3 One Zone-IA) -Übergang in Frage kommt, entscheidet sich Amazon S3 für den Übergang zu S3 Glacier Flexible Retrieval.

Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht.

So finden Sie heraus, wann Objekte ablaufen

Wenn Sie feststellen möchten, wann ein Objekt planmäßig abläuft, verwenden Sie die API-Operation HEAD Object oder GET Object. Diese API-Operationen geben Antwort-Header zurück, die das Datum und die Uhrzeit angeben, ab wann das Objekt nicht mehr zwischengespeichert werden kann.

Note

- Es kann eine Verzögerung zwischen dem Ablaufdatum und dem Datum geben, an dem Amazon S3 ein Objekt entfernt. Der Ablauf oder die mit einem abgelaufenen Objekt verbundene Speicherdauer wird Ihnen nicht in Rechnung gestellt.
- Verwenden Sie vor dem Aktualisieren, Deaktivieren oder Löschen von Lebenszyklusregeln die `LIST`-API-Operationen (z. B. ListObjectsV2ListObjectVersions, und

ListMultipartUploads) oder , Amazon S3 Inventory um zu überprüfen, ob Amazon S3 basierend auf Ihren Anwendungsfällen berechtigte Objekte übertragen und abgelaufen hat.

Gebühren für Mindestspeicherdauer

Wenn Sie eine S3-Lebenszyklusablaufregel erstellen, mit der Objekte ablaufen, die sich seit weniger als 30 Tagen in den Speicherklassen S3 Standard-IA oder S3 One Zone-IA befinden, werden Ihnen Gebühren für 30 Tage in Rechnung gestellt. Wenn Sie eine Lebenszyklusablaufregel erstellen, mit der Objekte ablaufen, die weniger als 90 Tage im Speicher S3 Glacier Flexible Retrieval gewesen sind, werden Ihnen Gebühren für 90 Tage in Rechnung gestellt. Wenn Sie eine Lebenszyklusablaufregel erstellen, mit der Objekte ablaufen, die weniger als 180 Tage im S3 Glacier Deep Archive-Speicher gewesen sind, werden Ihnen Gebühren für 180 Tage in Rechnung gestellt.

Weitere Informationen finden Sie unter Amazon S3 – Preise.

Festlegen der Lebenszyklus-Konfiguration für einen Bucket

In diesem Abschnitt wird erläutert, wie Sie mithilfe von AWS-SDKs, der AWS CLI oder der Amazon-S3-Konsole eine S3-Lebenszyklus-Konfiguration für einen Bucket festlegen können. Weitere Informationen zur S3-Lebenszyklus-Konfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Mit Lebenszyklusregeln können Sie Aktionen definieren, die Amazon S3 während der Lebensdauer eines Objekts ausführen soll (z. B. die Überführung von Objekten in eine andere Speicherklasse, ihre Archivierung oder ihr Löschen nach einem bestimmten Zeitraum).

Bevor Sie eine Lebenszyklus-Konfiguration festlegen, beachten Sie Folgendes:

Verzögerung bei der Verbreitung

Wenn Sie eine S3-Lebenszyklus-Konfiguration zu einem Bucket hinzufügen, tritt eine gewisse Verzögerung ein, bis eine neue oder aktualisierte Lebenszyklus-Konfiguration vollständig auf alle Amazon-S3-Systeme verteilt ist. Rechnen Sie mit einer Verzögerung von einigen Minuten, bis die Konfiguration vollständig wirksam ist. Diese Verzögerung kann auch auftreten, wenn Sie eine S3-Lebenszyklus-Konfiguration löschen.

Deaktivieren oder Löschen von Lebenszyklusregeln

Wenn Sie eine Lebenszyklusregel deaktivieren oder löschen, stellt Amazon S3 nach einer kurzen Verzögerung die Planung neuer Objekte zur Löschung oder Übertragung ein. Die Planung aller bereits geplanten Objekte wird aufgehoben und sie werden nicht gelöscht oder überführt.

> **Note**
>
> Verwenden Sie vor dem Aktualisieren, Deaktivieren oder Löschen von Lebenszyklusregeln die `LIST`-API-Operationen (z. B. ListObjectsV2ListObjectVersions, und ListMultipartUploads) oder , Amazon S3 Inventory um zu überprüfen, ob Amazon S3 basierend auf Ihren Anwendungsfällen berechtigte Objekte übertragen und abgelaufen hat. Wenn Probleme beim Aktualisieren, Deaktivieren oder Löschen von Lebenszyklusregeln auftreten, finden Sie weitere Informationen unter Fehlerbehebung bei Problemen mit dem Amazon-S3-Lebenszyklus.

Bestehende und neue Objekte

Wenn Sie einem Bucket eine Lebenszyklus-Konfiguration hinzufügen, gelten die Konfigurationsregeln für vorhandene Objekte und für Objekte, die Sie später hinzufügen. z. B.: Wenn Sie heute eine Lebenszyklus-Konfiguration mit einer Ablaufaktion hinzufügen, die dazu führt, dass Objekte mit einem bestimmten Präfix 30 Tage nach ihrer Erstellung ablaufen, setzt Amazon S3 alle bestehenden Objekte, die mehr als 30 tage alt sind, in die Löschwarteschlange.

Änderungen bei der Abrechnung

Zwischen dem Zeitpunkt der Erfüllung der Lebenszyklus-Konfigurationsregeln und der dadurch ausgelösten Aktion kann eine Verzögerung eintreten. Buchhaltungsänderungen werden jedoch sofort nach der Erfüllung der Lebenszyklus-Konfigurationsregel durchgeführt, auch wenn die Maßnahme noch nicht durchgeführt wurde.

Beispielsweise wird Ihnen nach dem Ablaufzeitpunkt des Objekts keine Speichergebühr in Rechnung gestellt, auch wenn das Objekt nicht sofort gelöscht wird. Ein weiteres Beispiel: Sobald die Objektübergangszeit abgelaufen ist, werden Ihnen Speichergebühren von S3 Glacier Flexible Retrieval in Rechnung gestellt, auch wenn das Objekt nicht sofort in die Speicherklasse S3 Glacier Flexible Retrieval überführt wird. Lebenszyklusübergänge auf die S3-Intelligent-Tiering-Speicherklasse sind die Ausnahme. Änderungen bei der Abrechnung treten erst auf, wenn das Objekt in die S3-Intelligent-Tiering-Speicherklasse übergegangen ist.

Verwenden der S3-Konsole

Sie können für alle Objekte oder eine Teilmenge der Objekte in einem Bucket Lebenszyklusregeln definieren, indem Sie ein gemeinsames Präfix (Objektnamen, die mit einer gemeinsamen Zeichenfolge beginnen) oder einen Tag verwenden. Mit einer Lebenszyklusregel können Sie spezifische Aktionen für aktuelle und nicht aktuelle Objektversionen definieren. Weitere Informationen finden Sie unter:

- Verwalten Ihres Speicher-Lebenszyklus
- Verwenden der Versioning in S3-Buckets

So erstellen Sie eine Lebenszyklusregel:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie eine Lebenszyklusregel erstellen möchten.
3. Wählen Sie den Tab Management (Verwaltung) und dann die Option Create lifecycle rule (Lebenszyklusregel erstellen).
4. Geben Sie unter Lifecycle rule name (Name der Lebenszyklusregel) einen Namen für Ihre Regel ein.

   Der Name muss innerhalb des Buckets eindeutig sein.
5. Wählen Sie den Umfang der Lebenszyklusregel:

   - Um diese Lebenszyklusregel auf alle Objekte mit einem bestimmten Präfix oder Tag anzuwenden, wählen Sie Umfang auf bestimmte Präfixe oder Markierungen beschränken aus.
     - Um den Bereich nach Präfix zu beschränken, geben Sie unter Prefix (Präfix) das Präfix ein.
     - Um den Bereich nach Tag einzuschränken, wählen Sie Add tag (Tag hinzufügen), und geben Sie den Tag-Schlüssel und den Wert ein.

     Weitere Hinweise zu Präfixen für Objektnamen finden Sie unter Erstellen von Objektschlüsselnamen. Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.
   - Um diese Lebenszyklusregel auf alle Objekte im Bucket anzuwenden, wählen Sie This rule applies to all objects in the bucket (Diese Regel gilt für alle Objekte in dem Bucket) und wählen

Sie I acknowledge that this rule applies to all objects in the bucket (Ich bestätige, dass diese Regel für alle Objekte in dem Bucket gilt).

6. Um eine Regel nach Objektgröße zu filtern, können Sie Minimale Objektgröße angeben, Maximale Objektgröße angeben oder beide Optionen aktivieren.

   - Wenn Sie eine minimale Objektgröße oder eine maximale Objektgröße angeben, muss der Wert größer als 0 Byte und bis zu 5 TB sein. Sie können diesen Wert in Byte, KB, MB oder GB angeben.
   - Wenn Sie beides angeben, muss die maximale Objektgröße größer als die minimale Objektgröße sein.

7. Wählen Sie unter Lifecycle rule actions (Lebenszyklusregelaktionen) die Aktionen aus, die Ihre Lebenszyklusregel ausführen soll:

   - Umsetzung aktueller Versionen von Objekten zwischen Speicherklassen
   - Umsetzung früherer Versionen von Objekten zwischen Speicherklassen
   - Ablauf aktueller Versionen von Objekten
   - Dauerhaftes Löschen früherer Versionen von Objekten
   - Löschen abgelaufener Löschmarkierungen oder unvollständiger mehrteiliger Uploads

   Abhängig von den von Ihnen ausgewählten Aktionen werden verschiedene Optionen angezeigt.

8. So setzen Sie aktuelle Versionen von Objekten zwischen Speicherklassen unter Transition current versions of objects between storage classes (Umsetzen von aktuellen Versionen von Objekten zwischen Speicherklassen) um:

   a. Wählen Sie unter Storage class transitions (Speicherklassenumsetzungen) die Speicherklasse aus, zu der Sie wechseln möchten:

      - Standard-IA
      - Intelligent-Tiering
      - One Zone-IA
      - S3 Glacier Flexible Retrieval
      - Glacier Deep Archive

   b. Geben Sie im Feld Days after object creation (Tage nach der Objekterstellung) die Anzahl der Tage nach der Erstellung für die Umsetzung des Objekts ein.

Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen. Sie können Übergänge für aktuelle oder vorhergehende Objektversionen definieren oder sowohl für aktuelle als auch vorhergehende Versionen. Versioning ermöglicht Ihnen, mehrere Versionen eines Objekts in einem Bucket aufzubewahren. Weitere Informationen über die Versionssteuerung finden Sie unter Verwenden der S3-Konsole.

> **Important**
>
> Wenn Sie die Speicherklasse S3 Glacier Flexible Retrieval oder Glacier Deep Archive wählen, bleibt Ihr Objekt in Amazon S3. Sie können nicht direkt über den separaten Amazon-S3-Glacier-Service auf sie zugreifen. Weitere Informationen finden Sie unter Übergang von Objekten mit Amazon-S3-Lebenszyklus.

9. So setzen Sie nicht aktuelle Versionen von Objekten zwischen Speicherklassen unter Transition non-current versions of objects between storage classes (Umsetzen von nicht aktuellen Versionen von Objekten zwischen Speicherklassen) um:

   a. Wählen Sie unter Storage class transitions (Speicherklassenumsetzungen) die Speicherklasse aus, zu der Sie wechseln möchten:

      - Standard-IA
      - Intelligent-Tiering
      - One Zone-IA
      - S3 Glacier Flexible Retrieval
      - Glacier Deep Archive

   b. Geben Sie im Feld Days after object becomes non-current (Tage, nachdem das Objekt nicht mehr aktuell ist) die Anzahl der Tage nach der Erstellung für die Umsetzung des Objekts ein.

10. Um den Ablauf aktueller Versionen von Objekten auszulösen, geben Sie unter Expire current versions of objects (Ablauf aktueller Objektversionen) in Number of days after object creation (Anzahl der Tage nach der Objekterstellung) die Anzahl der Tage ein.

> **Important**
>
> In einem Bucket ohne Versionierung führt die Ablaufaktion dazu, dass Amazon S3 das Objekt dauerhaft entfernt. Weitere Informationen über die Lebenszyklus-Aktionen finden Sie unter Elemente, die Lebenszyklus-Aktionen beschreiben.

11. Um frühere Versionen von Objekten dauerhaft zu löschen, geben Sie unter Permanently delete previous versions of objects (Dauerhaftes Löschen nicht aktueller Versionen von Objekten) in Days after objects become noncurrent (Tage nach dem Erlöschen von Objekten) die Anzahl der Tage ein. Sie können die Anzahl der beizubehaltenden neueren Versionen optional angeben, indem Sie einen Wert unter Number of newer versions to retain (Anzahl der beizubehaltenden neueren Versionen) eingeben.
12. Wählen Sie unter Delete expired delete markers or incomplete multipart uploads (Abgelaufene Löschmarkierungen oder unvollständige mehrteilige Uploads löschen) Delete expired object delete markers (Abgelaufene Objektlöschmarken löschen) und Delete incomplete multipart uploads (Unvollständige mehrteilige Uploads löschen) aus. Geben Sie dann die Anzahl der Tage nach der mehrteiligen Upload-Initiierung ein, die Sie beenden und für die Sie unvollständige mehrteilige Uploads löschen möchten.

    Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.
13. Wählen Sie Create rule aus.

    Wenn die Regel keine Fehler enthält, aktiviert Amazon S3 sie, und Sie können sie im Tab Management (Verwaltung) unter Lifecycle rules (Lebenszyklusregeln) sehen.

Informationen zu CloudFormation Vorlagen und Beispielen finden Sie unter Arbeiten mit AWS CloudFormation Vorlagen und AWS::S3::Bucket im AWS CloudFormation -Benutzerhandbuch.

Verwendung der AWS CLI

Zur Verwaltung der S3-Lebenszyklus-Konfigurationen können Sie die folgenden AWS CLI-Befehle verwenden:

- `put-bucket-lifecycle-configuration`
- `get-bucket-lifecycle-configuration`
- `delete-bucket-lifecycle`

Weitere Informationen zum Einrichten der AWS CLI finden Sie unter Entwickeln mit Amazon S3 über die AWS CLI.

Beachten Sie, dass die Amazon-S3-Lebenszyklus-Konfiguration eine XML-Datei ist. Wenn Sie jedoch die AWS CLI verwenden, können Sie die XML nicht angeben. Sie müssen stattdessen das JSON-Format angeben. Nachfolgend finden Sie Beispiele für XML-Lebenszyklus-Konfigurationen und entsprechenden JSON-Code, den Sie in einem AWS CLI-Befehl angeben können:

Betrachten Sie das folgende Beispiel einer S3-Lebenszyklus-Konfiguration:

Example Beispiel 1

JSON

```
{
    "Rules": [
        {
            "Filter": {
                "Prefix": "documents/"
            },
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 365,
                    "StorageClass": "GLACIER"
                }
            ],
            "Expiration": {
                "Days": 3650
            },
            "ID": "ExampleRule"
        }
    ]
        }
```

XML

```
<LifecycleConfiguration>
    <Rule>
        <ID>ExampleRule</ID>
        <Filter>
           <Prefix>documents/</Prefix>
```

```
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>365</Days>
           <StorageClass>GLACIER</StorageClass>
        </Transition>
        <Expiration>
             <Days>3650</Days>
        </Expiration>
    </Rule>
</LifecycleConfiguration>
```

Example Beispiel 2

JSON

```
{
    "Rules": [
        {
            "ID": "id-1",
            "Filter": {
                "And": {
                    "Prefix": "myprefix",
                    "Tags": [
                        {
                            "Value": "mytagvalue1",
                            "Key": "mytagkey1"
                        },
                        {
                            "Value": "mytagvalue2",
                            "Key": "mytagkey2"
                        }
                    ]
                }
            },
            "Status": "Enabled",
            "Expiration": {
                "Days": 1
            }
        }
    ]
}
```

XML

```
<LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Rule>
        <ID>id-1</ID>
        <Expiration>
            <Days>1</Days>
        </Expiration>
        <Filter>
            <And>
                <Prefix>myprefix</Prefix>
                <Tag>
                    <Key>mytagkey1</Key>
                    <Value>mytagvalue1</Value>
                </Tag>
                <Tag>
                    <Key>mytagkey2</Key>
                    <Value>mytagvalue2</Value>
                </Tag>
            </And>
        </Filter>
        <Status>Enabled</Status>
    </Rule>
</LifecycleConfiguration>
```

Sie können den Befehl `put-bucket-lifecycle-configuration` wie folgt testen.

So testen Sie die Konfiguration

1. Speichern Sie die JSON-Lebenszyklus-Konfiguration in einer Datei (`lifecycle.json`).
2. Führen Sie den folgenden AWS CLI-Befehl aus, um die Lebenszyklus-Konfiguration auf Ihrem Bucket einzurichten.

```
$ aws s3api put-bucket-lifecycle-configuration  \
--bucket bucketname  \
--lifecycle-configuration file://lifecycle.json
```

3. Rufen Sie zur Prüfung die S3-Lebenszyklus-Konfiguration wie folgt mit dem `get-bucket-lifecycle-configuration` AWS CLI-Befehl ab.

```
$ aws s3api get-bucket-lifecycle-configuration  \
```

```
--bucket bucketname
```

4. Verwenden Sie zum Löschen der S3-Lebenszyklus-Konfiguration wie folgt den `delete-bucket-lifecycle` AWS CLI-Befehl.

```
aws s3api delete-bucket-lifecycle \
--bucket bucketname
```

Verwenden der AWS-SDKs

Java

Sie können das AWS SDK for Java verwenden, um die S3-Lebenszyklus-Konfiguration eines Buckets zu verwalten. Weitere Informationen zur Verwaltung einer S3-Lebenszyklus-Konfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

> **Note**
>
> Wenn Sie einem Bucket eine S3-Lebenszyklus-Konfiguration hinzufügen, ersetzt Amazon S3 die aktuelle Lebenszyklus-Konfiguration des Buckets, sofern vorhanden. Um eine Lebenszyklus-Konfiguration zu aktualisieren, rufen Sie sie ab, nehmen die gewünschten Änderungen vor und fügen die geänderte Konfiguration dann dem Bucket hinzu.

Das folgende Beispiel veranschaulicht, wie Sie mit AWS SDK for Java die Lebenszyklus-Konfiguration eines Buckets hinzufügen, aktualisieren und löschen. Das Beispiel erledigt Folgendes:

- Fügt eine Lebenszyklus-Konfiguration zu einem Bucket hinzu.
- Ruft die Lebenszyklus-Konfiguration und Updates durch Hinzufügen einer weiteren Regel ab.
- Fügt die abgeänderte Lebenszyklus-Konfiguration dem Bucket hinzu. Amazon S3 ersetzt die vorhandene Konfiguration.
- Ruft die Konfiguration erneut ab und überprüft, ob sie die richtige Anzahl von Regeln enthält, indem die Anzahl der Regeln ausgegeben wird.
- Löscht die Lebenszyklus-Konfiguration und überprüft, ob sie gelöscht wurde, indem versucht wird, sie erneut abzurufen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketLifecycleConfiguration;
import com.amazonaws.services.s3.model.BucketLifecycleConfiguration.Transition;
import com.amazonaws.services.s3.model.StorageClass;
import com.amazonaws.services.s3.model.Tag;
import com.amazonaws.services.s3.model.lifecycle.LifecycleAndOperator;
import com.amazonaws.services.s3.model.lifecycle.LifecycleFilter;
import com.amazonaws.services.s3.model.lifecycle.LifecyclePrefixPredicate;
import com.amazonaws.services.s3.model.lifecycle.LifecycleTagPredicate;

import java.io.IOException;
import java.util.Arrays;

public class LifecycleConfiguration {

        public static void main(String[] args) throws IOException {
                Regions clientRegion = Regions.DEFAULT_REGION;
                String bucketName = "*** Bucket name ***";

                // Create a rule to archive objects with the "glacierobjects/"
 prefix to Glacier
                // immediately.
                BucketLifecycleConfiguration.Rule rule1 = new
 BucketLifecycleConfiguration.Rule()
                                .withId("Archive immediately rule")
                                .withFilter(new LifecycleFilter(new
 LifecyclePrefixPredicate("glacierobjects/")))
                                .addTransition(new
 Transition().withDays(0).withStorageClass(StorageClass.Glacier))
                                .withStatus(BucketLifecycleConfiguration.ENABLED);

                // Create a rule to transition objects to the Standard-Infrequent
 Access storage
                // class
```

```
                // after 30 days, then to Glacier after 365 days. Amazon S3 will
delete the
                // objects after 3650 days.
                // The rule applies to all objects with the tag "archive" set to
"true".
                BucketLifecycleConfiguration.Rule rule2 = new
BucketLifecycleConfiguration.Rule()
                                .withId("Archive and then delete rule")
                                .withFilter(new LifecycleFilter(new
LifecycleTagPredicate(new Tag("archive", "true"))))
                                .addTransition(new Transition().withDays(30)

.withStorageClass(StorageClass.StandardInfrequentAccess))
                                .addTransition(new
Transition().withDays(365).withStorageClass(StorageClass.Glacier))
                                .withExpirationInDays(3650)
                                .withStatus(BucketLifecycleConfiguration.ENABLED);

                // Add the rules to a new BucketLifecycleConfiguration.
                BucketLifecycleConfiguration configuration = new
BucketLifecycleConfiguration()
                                .withRules(Arrays.asList(rule1, rule2));

                try {
                        AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                                        .withCredentials(new
ProfileCredentialsProvider())
                                        .withRegion(clientRegion)
                                        .build();

                        // Save the configuration.
                        s3Client.setBucketLifecycleConfiguration(bucketName,
configuration);

                        // Retrieve the configuration.
                        configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);

                        // Add a new rule with both a prefix predicate and a tag
predicate.
                        configuration.getRules().add(new
BucketLifecycleConfiguration.Rule().withId("NewRule")
                                        .withFilter(new LifecycleFilter(new
LifecycleAndOperator(
```

```
                                                        Arrays.asList(new
LifecyclePrefixPredicate("YearlyDocuments/"),
                                                                new
LifecycleTagPredicate(new Tag(

   "expire_after",

   "ten_years"))))))
                                        .withExpirationInDays(3650)

.withStatus(BucketLifecycleConfiguration.ENABLED));

                        // Save the configuration.
                        s3Client.setBucketLifecycleConfiguration(bucketName,
configuration);

                        // Retrieve the configuration.
                        configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);

                        // Verify that the configuration now has three rules.
                        configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);
                        System.out.println("Expected # of rules = 3; found: " +
configuration.getRules().size());

                        // Delete the configuration.
                        s3Client.deleteBucketLifecycleConfiguration(bucketName);

                        // Verify that the configuration has been deleted by
attempting to retrieve it.
                        configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);
                        String s = (configuration == null) ? "No configuration
found." : "Configuration found.";
                        System.out.println(s);
                } catch (AmazonServiceException e) {
                        // The call was transmitted successfully, but Amazon S3
couldn't process
                        // it, so it returned an error response.
                        e.printStackTrace();
                } catch (SdkClientException e) {
                        // Amazon S3 couldn't be contacted for a response, or the
client
```

```
                        // couldn't parse the response from Amazon S3.
                        e.printStackTrace();
                }
        }
}
```

.NET

Sie können das AWS SDK for .NET verwenden, um die S3-Lebenszyklus-Konfiguration eines Buckets zu verwalten. Weitere Informationen zur Verwaltung einer Lebenszyklus-Konfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

> **Note**
>
> Wenn Sie eine Lebenszyklus-Konfiguration hinzufügen, ersetzt Amazon S3 die vorhandene Konfiguration für den angegebenen Bucket. Um eine Konfiguration zu aktualisieren, müssen Sie zuerst die Lebenszyklus-Konfiguration abrufen, die Änderungen vornehmen und dann die geänderte Lebenszyklus-Konfiguration dem Bucket hinzufügen.

Das folgende Beispiel veranschaulicht, wie Sie mit AWS SDK for .NET die Lebenszyklus-Konfiguration eines Buckets hinzufügen, aktualisieren und löschen. Das Codebeispiel führt die folgenden Aufgaben durch:

- Fügt eine Lebenszyklus-Konfiguration zu einem Bucket hinzu.
- Ruft die Lebenszyklus-Konfiguration und Updates durch Hinzufügen einer weiteren Regel ab.
- Fügt die abgeänderte Lebenszyklus-Konfiguration dem Bucket hinzu. Amazon S3 ersetzt die vorhandene Lebenszyklus-Konfiguration.
- Ruft die Konfiguration erneut ab und überprüft sie durch Ausgabe der Anzahl von Regeln in der Konfiguration.
- Löscht die Lebenszyklus-Konfiguration und überprüft den Löschvorgang.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
```

```
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class LifecycleTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            AddUpdateDeleteLifecycleConfigAsync().Wait();
        }

        private static async Task AddUpdateDeleteLifecycleConfigAsync()
        {
            try
            {
                var lifeCycleConfiguration = new LifecycleConfiguration()
                {
                    Rules = new List<LifecycleRule>
                        {
                            new LifecycleRule
                            {
                                 Id = "Archive immediately rule",
                                 Filter = new LifecycleFilter()
                                 {
                                     LifecycleFilterPredicate = new
 LifecyclePrefixPredicate()
                                     {
                                         Prefix = "glacierobjects/"
                                     }
                                 },
                                 Status = LifecycleRuleStatus.Enabled,
                                 Transitions = new List<LifecycleTransition>
                                 {
                                      new LifecycleTransition
                                      {
```

```
                                        Days = 0,
                                        StorageClass = S3StorageClass.Glacier
                                    }
                                },
                            },
                            new LifecycleRule
                            {
                                 Id = "Archive and then delete rule",
                                  Filter = new LifecycleFilter()
                                 {
                                     LifecycleFilterPredicate = new
LifecyclePrefixPredicate()
                                     {
                                         Prefix = "projectdocs/"
                                     }
                                 },
                                 Status = LifecycleRuleStatus.Enabled,
                                 Transitions = new List<LifecycleTransition>
                                 {
                                      new LifecycleTransition
                                      {
                                           Days = 30,
                                           StorageClass =
S3StorageClass.StandardInfrequentAccess
                                      },
                                      new LifecycleTransition
                                      {
                                        Days = 365,
                                        StorageClass = S3StorageClass.Glacier
                                      }
                                 },
                                 Expiration = new LifecycleRuleExpiration()
                                 {
                                       Days = 3650
                                 }
                            }
                        }
                };

                // Add the configuration to the bucket.
                await AddExampleLifecycleConfigAsync(client,
lifeCycleConfiguration);

                // Retrieve an existing configuration.
```

```
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);

                // Add a new rule.
                lifeCycleConfiguration.Rules.Add(new LifecycleRule
                {
                    Id = "NewRule",
                    Filter = new LifecycleFilter()
                    {
                        LifecycleFilterPredicate = new LifecyclePrefixPredicate()
                        {
                            Prefix = "YearlyDocuments/"
                        }
                    },
                    Expiration = new LifecycleRuleExpiration()
                    {
                        Days = 3650
                    }
                });

                // Add the configuration to the bucket.
                await AddExampleLifecycleConfigAsync(client,
lifeCycleConfiguration);

                // Verify that there are now three rules.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);
                Console.WriteLine("Expected # of rulest=3; found:{0}",
lifeCycleConfiguration.Rules.Count);

                // Delete the configuration.
                await RemoveLifecycleConfigAsync(client);

                // Retrieve a nonexistent configuration.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);

            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing
an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
```

```
            }
        }

        static async Task AddExampleLifecycleConfigAsync(IAmazonS3 client,
 LifecycleConfiguration configuration)
        {

            PutLifecycleConfigurationRequest request = new
 PutLifecycleConfigurationRequest
            {
                BucketName = bucketName,
                Configuration = configuration
            };
            var response = await client.PutLifecycleConfigurationAsync(request);
        }

        static async Task<LifecycleConfiguration>
 RetrieveLifecycleConfigAsync(IAmazonS3 client)
        {
            GetLifecycleConfigurationRequest request = new
 GetLifecycleConfigurationRequest
            {
                BucketName = bucketName
            };
            var response = await client.GetLifecycleConfigurationAsync(request);
            var configuration = response.Configuration;
            return configuration;
        }

        static async Task RemoveLifecycleConfigAsync(IAmazonS3 client)
        {
            DeleteLifecycleConfigurationRequest request = new
 DeleteLifecycleConfigurationRequest
            {
                BucketName = bucketName
            };
            await client.DeleteLifecycleConfigurationAsync(request);
        }
    }
}
```

Ruby

Sie können die verwendenAWS SDK for Ruby, um die S3-Lebenszykluskonfiguration für einen Bucket mithilfe der Klasse AWS::S3::BucketLifecycleKonfiguration zu verwalten. Weitere Informationen zur Verwendung von AWS SDK for Ruby mit Amazon S3 finden Sie unter Verwenden von AWS SDK for Ruby – Version 3. Weitere Informationen zur Verwaltung einer Lebenszyklus-Konfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Verwenden der REST-API

In den folgenden Abschnitten der Amazon Simple Storage Service API-Referenz wird die REST API im Zusammenhang mit der S3 Lebenszykluskonfiguration beschrieben.

- PUT Bucket-Lebenszyklus
- GET Bucket-Lebenszyklus
- DELETE Bucket-Lebenszyklus

Lebenszyklus- und andere Bucket-Konfigurationen

Neben den S3-Lebenszyklus-Konfigurationen können Sie Ihrem Bucket auch weitere Konfigurationen zuordnen. In diesem Abschnitt wird erläutert, wie sich die S3-Lebenszyklus-Konfiguration auf andere Bucket-Konfigurationen auswirkt.

Lebenszyklus und Versioning

Sie können nicht versionsfähigen Buckets und versionsfähigen Buckets S3-Lebenszyklus-Konfigurationen hinzufügen. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Ein versionsfähiger Bucket behält eine aktuelle Objektversion sowie und null oder mehr langfristige Objektversionen bei. Sie können separate Lebenszyklusrichtlinie für aktuelle und nicht aktuelle Objektversionen definieren.

Weitere Informationen finden Sie unter Elemente der Lebenszyklus-Konfiguration.

Important

Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere S3-Lebenszyklus-Aktionen auszuführen sind. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:

- Das permanente Löschen hat Vorrang vor einem Übergang.
- Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
- Wenn ein Objekt sowohl für einen S3 Glacier Flexible Retrieval als auch für einen S3 Standard-IA (oder S3 One Zone-IA) -Übergang in Frage kommt, entscheidet sich Amazon S3 für den Übergang zu S3 Glacier Flexible Retrieval.

Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht.

Lebenszyklus-Konfiguration auf MFA-fähigen Buckets

Eine Lebenszyklus-Konfiguration wird auf MFA-fähigen Buckets (Multi-Factor Authentication) nicht unterstützt.

Lebenszyklus und Protokollieren

Amazon-S3-Lebenszyklusaktionen werden nicht von der Protokollierung auf AWS CloudTrail Objektebene erfasst. CloudTrail erfasst API-Anforderungen an externe Amazon S3-Endpunkte, während S3-Lebenszyklusaktionen mit internen Amazon S3-Endpunkten ausgeführt werden. Amazon-S3-Server-Zugriffsprotokolle können in einem S3-Bucket aktiviert werden, um S3-Lebenszyklus-Aktionen wie den Objektübergang zu einer anderen Speicherklasse und einen Objektablauf zu erfassen, was zu dauerhaftem Löschen oder logischen Löschen führt. Weitere Informationen finden Sie unter the section called “Protokollierungs-Serverzugriff”.

Wenn die Protokollierung für Ihren Bucket aktiviert ist, melden Amazon-S3-Server-Zugriffsprotokolle die Ergebnisse der folgenden Vorgänge.

Operationsprotokoll	Beschreibung
`S3.EXPIRE.OBJECT`	Amazon S3 löscht das Objekt aufgrund der Lebenszyklusablaufaktion permanent.

Operationsprotokoll	Beschreibung
S3.CREATE.DELETEMARKER	Amazon S3 löscht die aktuelle Version logisch und fügt eine Löschmarkierung in einem Bucket mit aktiviertem Versioning hinzu.
S3.TRANSITION_SIA.OBJECT	Amazon S3 überträgt das Objekt in die Speicherklasse S3 Standard-IA.
S3.TRANSITION_ZIA.OBJECT	Amazon S3 überträgt das Objekt in die Speicherklasse S3 One Zone-IA.
S3.TRANSITION_INT.OBJECT	Amazon S3 überträgt das Objekt in die Speicherklasse S3 Intelligent-Tiering.
S3.TRANSITION_GIR.OBJECT	Amazon S3 initiiert den Übergang von Objekten in die Speicherklasse S3 Glacier Instant Retrieval.
S3.TRANSITION.OBJECT	Amazon S3 initiiert den Übergang von Objekten in die Speicherklasse S3 Glacier Flexible Retrieval.
S3.TRANSITION_GDA.OBJECT	Amazon S3 initiiert den Übergang von Objekten in die Speicherklasse S3 Glacier Deep Archive.
S3.DELETE.UPLOAD	Amazon S3 bricht nicht vollständige mehrteilige Uploads ab.

Note

Amazon-S3-Server-Zugriffsprotokolle werden in der Regel auf Best-Effort-Basis bereitgestellt. Sie sind nicht als vollständige Auflistung aller Amazon-S3-Anforderungen vorgesehen.

Fehlerbehebung bei S3 Lifecycle

Weitere Informationen zur Behebung häufiger Probleme mit S3 Lifecycle finden Sie unter Fehlerbehebung bei Problemen mit dem Amazon-S3-Lebenszyklus.

Weitere Informationen

- Elemente der Lebenszyklus-Konfiguration
- Übergang in die Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive (Objektarchivierung)
- Festlegen der Lebenszyklus-Konfiguration für einen Bucket

Konfigurieren von Lebenszyklus-Ereignisbenachrichtigungen

Sie können eine Amazon-S3-Ereignisbenachrichtigung einrichten, um eine Benachrichtigung zu erhalten, wenn Amazon S3 ein Objekt löscht oder es nach einer S3-Lebenszyklusregel in eine andere Amazon-S3-Speicherklasse überführt.

Bei Verwendung der LifecycleExpiration-Ereignistypen können Sie Benachrichtigungen erhalten, sobald Amazon S3 ein Objekt basierend auf Ihrer S3-Lebenszyklus-Konfiguration löscht. Der Ereignistyp `s3:LifecycleExpiration:Delete` benachrichtigt Sie, wenn ein Objekt in einem unversionierten Bucket gelöscht wird. Es benachrichtigt Sie auch, wenn eine Objektversion durch eine S3-Lebenszyklus-Konfiguration dauerhaft gelöscht wird. Der Ereignistyp `s3:LifecycleExpiration:DeleteMarkerCreated` benachrichtigt Sie, wenn S3 Lebenszyklus eine Löschmarke erstellt, wenn eine aktuelle Version eines Objekts im versionierten Bucket gelöscht wird. Weitere Informationen hierzu finden Sie unter Objektversion löschen.

Durch Verwendung des Ereignistyps s3:LifecycleTransition können Sie eine Benachrichtigung erhalten, wenn ein Objekt von einer Amazon-S3-Speicherklasse in eine andere durch eine S3-Lebenszyklus-Konfiguration überführt wird.

Amazon S3 kann Ereignisbenachrichtigungen in einem Amazon Simple Notification Service (Amazon SNS)-Thema, einer Amazon Simple Queue Service (Amazon SQS)-Warteschlange oder einer AWS Lambda-Funktion veröffentlichen. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Anweisungen zum Konfigurieren von Amazon-S3-Ereignisbenachrichtigungen finden Sie unter Aktivieren von Ereignisbenachrichtigungen.

Das Folgende ist ein Beispiel für eine Nachricht, die Amazon S3 sendet, um ein s3:LifecycleExpiration:Delete-Ereignis zu veröffentlichen. Weitere Informationen finden Sie unter Struktur der Ereignisnachricht.


```
{
   "Records":[
      {
         "eventVersion":"2.3",
         "eventSource":"aws:s3",
         "awsRegion":"us-west-2",
         "eventTime":"1970-01-01T00:00:00.000Z",
         "eventName":"LifecycleExpiration:Delete",
         "userIdentity":{
            "principalId":"s3.amazonaws.com"
         },
         "requestParameters":{
            "sourceIPAddress":"s3.amazonaws.com"
         },
         "responseElements":{
            "x-amz-request-id":"C3D13FE58DE4C810",
            "x-amz-id-2":"FMyUVURIY8/IgAtTv8xRjskZQpcIZ9KG4V5Wp6S7S/
JRWeUWerMUE5JgHvANOjpD"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"testConfigRule",
            "bucket":{
               "name":"mybucket",
               "ownerIdentity":{
                  "principalId":"A3NL1KOZZKExample"
               },
               "arn":"arn:aws:s3:::mybucket"
            },
            "object":{
               "key":"expiration/delete",
               "sequencer":"0055AED6DCD90281E5",
            }
         }
      }
   ]
}
```

Nachrichten, die Amazon S3 sendet, um ein s3:LifecycleTransition-Ereignis zu veröffentlichen, enthalten auch die folgenden Informationen.

```
"lifecycleEventData":{
    "transitionEventData": {
        "destinationStorageClass": the destination storage class for the object
    }
}
```

Elemente der Lebenszyklus-Konfiguration

Themen

- ID-Element
- Statuselement
- Filterelement
- Elemente, die Lebenszyklus-Aktionen beschreiben

Sie erstellen eine S3-Lebenszyklus-Konfiguration als XML, die aus einen oder mehreren Lebenszyklusregeln besteht.

```
<LifecycleConfiguration>
    <Rule>
         ...
    </Rule>
    <Rule>
         ...
    </Rule>
</LifecycleConfiguration>
```

Jeder Regel umfasst Folgendes:

- Metadaten für die Regel, mit Regel-ID und einem Status, der anzeigt, ob die Regel aktiviert oder deaktiviert ist. Wenn eine Regel deaktiviert ist, führt Amazon S3 keine in der Regel spezifizierten Aktionen aus.
- Filter, der die Objekte identifiziert, für die die Regel gilt. Sie können einen Filter angeben, indem Sie die Objektgröße, das Objektschlüsselpräfix, ein oder mehrere Objekt-Tags oder eine Kombination von Filtern verwenden.

- Eine oder mehrere Übergangs- oder Ablaufaktionen mit einem Datum oder einem Zeitintervall innerhalb der Lebensdauer des Objekts, zu denen Amazon S3 die angegebene Aktion ausführen soll.

In den folgenden Abschnitten werden die XML-Elemente in einer S3-Lebenszyklus-Konfiguration beschrieben. Beispielkonfigurationen finden Sie unter Beispiele der S3-Lebenszyklus-Konfiguration.

ID-Element

Eine S3-Lebenszyklus-Konfiguration kann bis zu 1.000 Regeln haben. Diese Grenze ist nicht einstellbar. Das Element **<ID>** identifiziert eine Regel eindeutig. Die Länge der ID ist auf 255 Zeichen begrenzt.

Statuselement

Der Wert des Elements **<Status>** kann „Enabled (Aktiviert)“ oder „Disabled (Deaktiviert)“ sein. Wenn eine Regel deaktiviert ist, führt Amazon S3 keine in der Regel definierten Aktionen aus.

Filterelement

Eine Lebenszyklusregel kann für alle Objekte oder eine Untermenge der Objekte in einem Bucket gelten, abhängig vom Element **<Filter>**, das Sie in der Lebenszyklusregel angeben.

Sie können Objekte nach dem Schlüsselpräfix, Objekt-Markierungen oder Kombinationen aus beidem filtern (bei einer Kombination verwendet Amazon S3 ein logisches `AND` für die Filter). Betrachten Sie die folgenden Beispiele:

- Angabe eines Filters unter Verwendung von Schlüsselpräfixen – Dieses Beispiel zeigt eine S3-Lebenszyklusregel, die sich abhängig vom Schlüsselnamenpräfix auf eine Untermenge von Objekten bezieht (`logs/`). Beispielsweise gilt die Lebenszyklusregel für die Objekte `logs/mylog.txt`, `logs/temp1.txt` und `logs/test.txt`. Die Regel gilt nicht für das Objekt `example.jpg`.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Prefix>logs/</Prefix>
        </Filter>
        transition/expiration actions.
         ...
```

```
    </Rule>
    ...
</LifecycleConfiguration>
```

Wenn Sie eine Lebenszyklus-Aktion auf eine Untermenge von Objekten basierend auf unterschiedlichen Schlüsselnamenpräfixen anwenden wollen, müssen Sie separate Regeln angeben. Geben Sie in jeder Regel einen auf einem Präfix basierenden Filter an. Um beispielsweise eine Lebenszyklus-Aktion für Objekte mit den Schlüsselpräfixen `projectA/` und `projectB/` zu beschreiben, geben Sie zwei Regeln an, die wie folgt aussehen.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Prefix>projectA/</Prefix>
        </Filter>
        transition/expiration actions.
         ...
    </Rule>

    <Rule>
        <Filter>
           <Prefix>projectB/</Prefix>
        </Filter>
        transition/expiration actions.
         ...
    </Rule>
</LifecycleConfiguration>
```

Weitere Informationen über Objektschlüssel finden Sie unter Erstellen von Objektschlüsselnamen.

- Angabe eines Filters auf der Basis von Objekt-Markierungen – Im folgenden Beispiel gibt die Lebenszyklusregel einen Filter basierend auf einem Tag (*key*) und einem Wert (*value*) an. Die Regel wird dann nur auf eine Untermenge von Objekten mit dem spezifischen Tag angewendet.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Tag>
              <Key>key</Key>
              <Value>value</Value>
           </Tag>
```

```
        </Filter>
        transition/expiration actions.
        ...
    </Rule>
</LifecycleConfiguration>
```

Sie können einen Filter auf mehreren Markierungen basierend angeben. Sie müssen die Markierungen mit dem Element <And> umschließen wie im folgenden Beispiel gezeigt. Die Regel weist Amazon S3 an, Lebenszyklus-Aktionen für Objekte mit zwei Markierungen auszuführen (mit dem spezifischen Tag-Schlüssel und -wert).

```
<LifecycleConfiguration>
    <Rule>
      <Filter>
         <And>
            <Tag>
               <Key>key1</Key>
               <Value>value1</Value>
            </Tag>
            <Tag>
               <Key>key2</Key>
               <Value>value2</Value>
            </Tag>
             ...
          </And>
      </Filter>
      transition/expiration actions.
    </Rule>
</Lifecycle>
```

Die Lebenszyklusregel gilt für Objekte, für die beide Markierungen angegeben sind. Amazon S3 führt eine logische AND-Operation aus. Beachten Sie Folgendes:

- Jedes Tag muss exakt mit dem Schlüssel und dem Wert übereinstimmen.
- Die Regel gilt für die Untermenge der Objekte, , die alle in der Regel angegebenen Markierungen besitzt. Wenn für ein Objekt zusätzliche Markierungen angegeben sind, gilt die Regel weiterhin.

> **Note**
>
> Wenn Sie mehrere Markierungen in einem Filter spezifizieren, muss jeder Tag-Schlüssel eindeutig sein.

- Angabe eines Filters auf der Basis eines Präfixes und mindestens eines Markierungen – Sie können in einer Lebenszyklusregel einen Filter angeben, der sowohl auf dem Schlüsselpräfix als auch auf mindestens einem Tag basiert. Auch hier müssen Sie all diese mit dem Element <And> umschließen, wie im Folgenden gezeigt.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
          <And>
             <Prefix>key-prefix</Prefix>
             <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
             </Tag>
             <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
             </Tag>
              ...
          </And>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions.
    </Rule>
</LifecycleConfiguration>
```

  Amazon S3 kombiniert diese Filter unter Verwendung einer logischen AND-Operation. Das bedeutet, die Regel wird auf eine Untermenge von Objekten mit einem spezifischen Schlüsselpräfix und spezifischen Tag angewendet. Ein Filter kann höchstens ein Präfix und null oder mehr Markierungen aufweisen.

- Sie können einen leeren Filter angeben, dann gilt die Regel für alle Objekte in dem Bucket.

```
<LifecycleConfiguration>
    <Rule>
```

```
        <Filter>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions.
    </Rule>
</LifecycleConfiguration>
```

- Um eine Regel nach Objektgröße zu filtern, können Sie eine Mindestgröße (`ObjectSizeGreaterThan`) oder eine Höchstgröße (`ObjectSizeLessThan`) oder einen Bereich von Objektgrößen angeben.

  Objektgrößenwerte sind in Bytes. Die maximale Filtergröße beträgt 5 TB. Für einige Speicherklassen gelten Beschränkungen für die Mindestobjektgröße. Weitere Informationen finden Sie unter Vergleich der Amazon-S3-Speicherklassen.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
            <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions.
    </Rule>
</LifecycleConfiguration>
```

  Wenn Sie einen Objektgrößenbereich angeben, muss die `ObjectSizeGreaterThan`-Ganzzahl kleiner als der `ObjectSizeLessThan`-Wert sein. Wenn Sie mehr als einen Filter verwenden, müssen Sie die Filter in ein `<And>`-Element packen. Im folgenden Beispiel wird gezeigt, wie Sie Objekte in einem Bereich zwischen 500 und 64000 Byte angeben.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
            <And>
                <Prefix>key-prefix</Prefix>
                <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
                <ObjectSizeLessThan>64000</ObjectSizeLessThan>
            </And>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions.
```

```
    </Rule>
</LifecycleConfiguration>
```

Elemente, die Lebenszyklus-Aktionen beschreiben

Sie können Amazon S3 anweisen, spezifische Aktionen innerhalb der Lebensdauer eines Objekts auszuführen, indem Sie eine oder mehrere vordefinierte Aktionen in einer S3-Lebenszyklusregel angeben. Die Wirkung dieser Aktionen ist vom Versioning-Status Ihres Buckets abhängig.

- Transition-Aktionselement – Sie geben die `Transition`-Aktion an, um Objekte von einer Speicherklasse in eine andere zu überführen. Weitere Informationen zum Übergang von Objekten finden Sie unter Unterstützte Transaktionen und zugehörige Einschränkungen. Wenn ein vorgegebenes Datum oder einem Zeitintervall innerhalb der Lebensdauer des Objekts erreicht ist, führt Amazon S3 den Übergang aus.

  Für einen versionsfähigen Bucket (Bucket mit aktiviertem oder ausgesetztem Versioning) wird die `Transition`-Aktion auf die aktuelle Objektversion angewendet. Um nicht aktuelle Versionen zu verwalten, definiert Amazon S3 die `NoncurrentVersionTransition`-Aktion (wird später in diesem Thema beschrieben).

- Ablaufaktions-Element – Die `Expiration`-Aktion lässt in der Regel definierte Objekte ablaufen und gilt für entsprechende Objekte in einer der Amazon-S3-Speicherklassen. Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen. Amazon S3 lässt keinen Zugriff auf abgelaufene Objekte zu. Ob die Objekte dauerhaft entfernt werden, ist vom Versioning-Status des Buckets abhängig.
  - Nicht versionsfähiger Bucket – Die `Expiration`-Aktion bewirkt, dass Amazon S3 das Objekt dauerhaft entfernt.
  - Versionsfähiger Bucket – Für einen versionsfähigen Bucket (d. h. das Versioning ist aktiviert oder ausgesetzt) gibt es mehrere Aspekte, die bestimmen, wie Amazon S3 `Expiration`-Aktion verarbeitet. Für Buckets mit aktivierter oder ausgesetzter Versionsverwaltung gilt Folgendes:
    - Die `Expiration`-Aktion wird nur auf die aktuelle Version angewendet (sie wirkt sich nicht auf nicht aktuelle Objektversionen aus).
    - Amazon S3 führt keine Aktion aus, wenn es eine oder mehrere Objektversionen gibt und die Löschmarkierung die aktuelle Version ist.
    - Wenn die aktuelle Objektversion die einzige Objektversion und auch eine Löschmarkierung ist (auch als Löschmarkierung eines abgelaufenen Objekts bezeichnet, wobei alle

Objektversionen gelöscht werden und nur noch eine Löschmarkierung beibehalten wird), entfernt Amazon S3 die Löschmarkierung des abgelaufenen Objekts. Sie können die Ablaufaktion auch verwenden, um Amazon S3 anzuweisen, alle abgelaufenen Löschmarkierungen zu entfernen. Ein Beispiel finden Sie unter Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte.

Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Berücksichtigen Sie auch die folgenden Punkte, wenn Sie Amazon S3 zur Ablaufverwaltung einrichten:

- Bucket mit aktiviertem Versioning

  Wenn die aktuelle Objektversion keine Löschmarkierung ist, fügt Amazon S3 die Löschmarkierung mit einer eindeutigen Versions-ID hinzu. Damit ist die aktuelle Version nicht mehr aktuell und die Löschmarkierung wird zur aktuellen Version.

- Bucket mit ausgesetztem Versioning

  Für einen Bucket mit ausgesetztem Versioning bewirkt die Ablaufaktion, dass Amazon S3 eine Löschmarkierung mit der Versions-ID null erstellt. Diese Löschmarkierung ersetzt jede Objektversion mi einer Versions-ID von null in der Versionshierarchie, womit das Objekt effektiv gelöscht wird.

Darüber hinaus unterstützt Amazon S3 die folgenden Aktionen, mit denen Sie nicht aktuelle Objektversionen in einem versionsfähigen Bucket verwalten können (d. h. für Buckets mit aktivierten und ausgesetztem Versioning).

- NoncurrentVersionTransition Aktionselement – Verwenden Sie diese Aktion, um anzugeben, wann Amazon S3 Objekte in die angegebene Speicherklasse überführen soll. Sie können diesen Ablauf auf einer bestimmten Anzahl von Tagen, seit der die Objekte nicht aktuell sind, basieren. Neben der Anzahl der Tage können Sie auch eine maximale Anzahl von nicht aktuellen Versionen angeben, die beibehalten werden müssen. Dieser Wert bestimmt, wie viele neuere nicht aktuelle Versionen vorhanden sein müssen, bevor Amazon S3 die zugehörige Aktion für eine bestimmte Version ausführen kann. Sie müssen auch ein `Filter`-Element angeben, um die maximale Anzahl nicht aktueller Versionen anzugeben. Wenn Sie kein `Filter`-Element angeben, generiert Amazon S3 einen `InvalidRequest`-Fehler, sobald Sie eine maximale Anzahl nicht aktueller Versionen angeben.

Weitere Informationen zum Übergang von Objekten finden Sie unter Unterstützte Transaktionen und zugehörige Einschränkungen. Ausführliche Informationen dazu, wie Amazon S3 das Datum berechnet, wenn Sie die Anzahl der Tage in der `NoncurrentVersionTransition`-Aktion angeben, finden Sie unter Lebenszyklusregeln: Basierend auf dem Alter eines Objekts.

- NoncurrentVersionExpiration -Aktionselement – Verwenden Sie diese Aktion, um anzugeben, dass Amazon S3 nicht aktuelle Versionen von Objekten dauerhaft löschen soll. Diese gelöschten Objekte können nicht wiederhergestellt werden. Sie können diesen Ablauf auf einer bestimmten Anzahl von Tagen, seit der die Objekte nicht aktuell sind, basieren. Neben der Anzahl der Tage können Sie auch eine maximale Anzahl von nicht aktuellen Versionen angeben, die beibehalten werden müssen. Dieser Wert gibt die Anzahl der neueren nicht aktuellen Versionen an, die vorhanden sein müssen, damit Amazon S3 die zugehörige Aktion für eine bestimmte Version ausführen kann. Sie müssen auch ein `Filter`-Element angeben, um die maximale Anzahl nicht aktueller Versionen anzugeben. Wenn Sie kein `Filter`-Element angeben, generiert Amazon S3 einen `InvalidRequest`-Fehler, sobald Sie eine maximale Anzahl nicht aktueller Versionen angeben.

  Das verzögerte Entfernen nicht aktueller Objekte kann hilfreich sein, wenn Sie versehentliche Lösch- oder Überschreibvorgänge korrigieren müssen. Beispielsweise können Sie eine Ablaufregel konfigurieren, um nicht aktuelle Versionen fünf Tage nach dem Zeitpunkt zu löschen, zu dem sie nicht aktuell geworden sind. Angenommen, Sie erstellen am 01.01.2014 um 10:30 AM UTC das Objekt `photo.gif` (Versions-ID 111111). Am 2.1.2014 um 11:30 AM UTC löschen Sie versehentlich `photo.gif` (Versions-ID 111111), wodurch eine Löschmarkierung mit einer neuen Versions-ID erstellt wird (z. B. Versions-ID 4857693). Jetzt haben Sie fünf Tage Zeit, die Originalversion von `photo.gif` (Versions-ID 111111) wiederherzustellen, bevor das Löschen permanent wird. Am 8.1.2014 um 00:00 UTC wird die Lebenszyklusregel für den Ablauf ausgeführt und löscht `photo.gif` (Versions-ID 111111) permanent. Dies erfolgt fünf Tage, nachdem es zu einer nicht aktuellen Version geworden ist.

  Weitere Informationen darüber, wie Amazon S3 das Datum berechnet, an dem Sie die Anzahl der Tage `NoncurrentVersionExpiration` Aktion angeben finden Sie in Lebenszyklusregeln: Basierend auf dem Alter eines Objekts.

  > **Note**
  >
  > Objektablauf-Lebenszykluskonfigurationen entfernen keine unvollständigen mehrteiligen Uploads. Um unvollständige mehrteilige Uploads zu entfernen, müssen Sie die

AbortIncompleteMultipartUpload Lebenszyklus-Konfigurationsaktion verwenden, die später in diesem Abschnitt beschrieben wird.

Neben den Übergangs- und Ablaufaktionen können Sie die folgende Lebenszyklus-Konfigurationsaktion verwenden, um Amazon S3 anzuweisen, unvollständige mehrteilige Uploads abzubrechen.

- AbortIncompleteMultipartUpload Aktionselement – Verwenden Sie dieses Element, um eine maximale Zeit (in Tagen) festzulegen, für die mehrteilige Uploads ausgeführt werden sollen. Wenn die jeweiligen mehrteiligen Uploads (festgelegt durch das in der Lebenszyklusregel angegebene `prefix` des Schlüsselnamens) nicht innerhalb des vordefinierten Zeitraums erfolgreich abgeschlossen werden, bricht Amazon S3 die unvollständigen mehrteiligen Uploads ab. Weitere Informationen finden Sie unter Abbrechen eines mehrteiligen Uploads.

  > **Note**
  >
  > Sie können diese Lebenszyklus-Aktion nicht in einer Regel angeben, die einen auf Objekt-Markierungen basierten Filter verwendet.

- ExpiredObjectDeleteMarker Aktionselement – In einem Bucket mit aktiviertem Versioning wird eine Löschmarkierung mit null nicht aktuellen Versionen als Löschmarkierung für abgelaufene Objekte bezeichnet. Sie können diese Lebenszyklus-Aktion verwenden, um S3 anzuweisen, die Löschmarkierungen der abgelaufenen Objekte zu entfernen. Ein Beispiel finden Sie unter Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte.

  > **Note**
  >
  > Sie können diese Lebenszyklus-Aktion nicht in einer Regel angeben, die einen auf Objekt-Markierungen basierten Filter verwendet.

Wie Amazon S3 berechnet, wie lange ein Objekt nicht aktuell ist

In einem versionsfähigen Bucket können Sie mehrere Versionen eines Objekts haben. Es gibt immer eine aktuelle Version und null oder mehr nicht aktuelle Versionen. Immer wenn Sie ein Objekt hochladen, wird die aktuelle Version als die nicht aktuelle Version beibehalten, und die

neu hinzugefügte Version, der Nachfolger, wird zur aktuellen Version. Um die Anzahl der Tage zu bestimmen, wie lange ein Objekt nicht aktuell ist, wertet Amazon S3 es aus, wenn der Nachfolger erstellt wurde. Amazon S3 verwendet die Anzahl der Tage, seit der Nachfolger erstellt wurde, als die Anzahl der Tage, wie lange ein Objekt nicht aktuell ist.

Wiederherstellung vorheriger Versionen eines Objekts bei Verwendung von S3-Lebenszyklus-Konfigurationen

Wie im Thema Wiederherstellen früherer Versionen detailliert erklärt, können Sie eine der beiden folgenden Methoden verwenden, um vorherige Versionen eines Objekts abzurufen:

1. Durch Kopieren einer nicht aktuellen Version des Objekts in denselben Bucket. Das kopierte Objekt wird zur aktuellen Version dieses Objekts, und alle Objektversionen werden beibehalten.
2. Durch das dauerhafte Löschen der aktuellen Version des Objekts. Wenn Sie die aktuelle Objektversion löschen, wandeln Sie letztlich die nicht aktuelle Version in die aktuelle Version dieses Objekts um.

Wenn Sie S3-Lebenszyklus-Konfigurationsregeln für Buckets mit aktiviertem Versioning verwenden, empfehlen wir Ihnen, die erste Methode zu verwenden, was sich bewährt hat. Der S3-Lebenszyklus arbeitet nach einem letztlich konsistenten Modell. Eine aktuelle Version, die Sie dauerhaft gelöscht haben, verschwindet möglicherweise erst, wenn die Änderungen propagiert werden (Amazon S3 kennt diese Löschoperation möglicherweise nicht). In der Zwischenzeit kann die Lebenszyklusregel, die Sie für den Ablauf nicht aktueller Objekte konfiguriert haben, die nicht aktuellen Objekte dauerhaft entfernen, auch dasjenige, das Sie wiederherstellen möchten. Das Kopieren der alten Version, wie in der ersten Methode empfohlen, ist deshalb die sicherere Alternative.

Lebenszyklusregeln: Basierend auf dem Alter eines Objekts

Sie können ein Zeitintervall als Anzahl der Tage ab der Erstellung (oder Änderung) der Objekte angeben, wann Amazon S3 die Aktion ausführen kann.

Wenn Sie die Anzahl der Tage in den `Transition`- und `Expiration`-Aktionen in einer S3-Lebenszyklus-Konfiguration angeben, beachten Sie Folgendes:

- Dies ist die Anzahl der Tage nach der Erstellung des Objekts, wann die Aktion stattfindet.

- Amazon S3 berechnet die Zeit, indem es die in der Regel angegebene Anzahl an Tagen zur Zeit der Objekterstellung hinzufügt und die resultierende Zeit auf die UTC des nächsten Markierungen um Mitternacht rundet. Wurde ein Objekt beispielsweise am 15.1.2014 um 10:30 AM UTC erstellt und Sie geben in einer Übergangsregel 3 Tage an, wird das Übergangsdatum des Objekts für den 19.1.2014 um 00:00 UTC berechnet.

> **Note**
>
> Amazon S3 behält nur das letzte Änderungsdatum für jedes Objekt bei. Beispielsweise zeigt die Amazon-S3-Konsole das Datum Last Modified (Zuletzt geändert) im Bereich Properties (Eigenschaften) für das Objekt an. Wenn Sie ein neues Objekt erstellen, ist dieses Datum das Datum, zu dem das Objekt erstellt wurde. Wenn Sie das Objekt ersetzen, ändert sich das Datum entsprechend. Der Begriff Erstellungsdatum ist daher gleichbedeutend mit dem Begriff letztes Änderungsdatum.

Wenn Sie die Anzahl der Tage in den `NoncurrentVersionTransition`- und `NoncurrentVersionExpiration`-Aktionen in einer Lebenszyklus-Konfiguration angeben, beachten Sie Folgendes:

- Dies ist die Anzahl der Tage, ab dem Zeitpunkt, an dem die Version des Objekts nicht aktuell wird (d. h. der Zeitpunkt, an dem das Objekt überschrieben oder gelöscht wird). Dies gilt für die Version des Objekts, für das Amazon S3 die Aktion ausführen wird.
- Amazon S3 berechnet die Zeit, indem es die in der Regel angegebene Anzahl an Tagen der Zeit hinzufügt, zu der die neue Nachfolgerversion des Objekts erstellt wurde, und die resultierende Zeit auf die UTC des nächsten Markierungen um Mitternacht rundet. Angenommen, Sie haben in Ihrem Bucket eine aktuelle Version eines Objekts, das am 1.1.2014 um 10:30 AM UTC erstellt wurde. Wenn die neue Version des Objekts, die die aktuelle Version ersetzt, am 15.1.2014 um 10:30 AM UTC erstellt wird und Sie in einer Übergangsregel 3 Tage angeben, wird das Übergangsdatum für das Objekt für den 19.1.2014 um 00:00 UTC berechnet.

Lebenszyklusregeln: Basierend auf einem spezifischen Datum

Wenn Sie in einer S3-Lebenszyklus-Regel eine Aktion angeben, können Sie ein Datum angeben, wann Amazon S3 die Aktion ausführen soll. Wenn das spezifische Datum erreicht ist, wendet Amazon S3 die Aktion auf alle qualifizierten Objekte an (basierend auf den Filterkriterien).

Wenn Sie eine S3-Lebenszyklus-Aktion mit einem Datum angeben, das in der Vergangenheit liegt, kommen sofort alle qualifizierten Objekte für diese Lebenszyklus-Aktion in Frage.

> **Important**
>
> Die auf einem Datum basierende Aktion ist keine einmalige Aktion. Amazon S3 wendet die auf dem Datum basierende Aktion auch an, nachdem das Datum erreicht wurde, solange der Regel-Status is `Enabled`.
> Angenommen, Sie geben eine auf einem Datum basierende `Expiration`-Aktion an, um alle Objekte zu löschen (unter der Annahme, dass in der Regel kein Filter angegeben ist). Amazon S3 lässt zu dem angegebenen Datum alle Objekte in dem Bucket ablaufen. S3 lässt auch weiterhin alle neuen Objekte ablaufen, die Sie in dem Bucket erstellen. Um die Lebenszyklus-Aktion zu unterbrechen, müssen Sie die Aktion aus der Lebenszyklus-Konfiguration entfernen, die Regel deaktivieren oder die Regel aus der Lebenszyklus-Konfiguration löschen.

Der Datumswert muss konform zum Format ISO 8601 angegeben werden. Die Uhrzeit ist stets Mitternacht UTC.

> **Note**
>
> Sie können die auf dem Datum basierenden Lebenszyklusregeln nicht über die Amazon-S3-Konsole erstellen, aber Sie können solche Regeln anzeigen, deaktivieren oder löschen.

Beispiele der S3-Lebenszyklus-Konfiguration

Dieser Abschnitt enthält Beispiele für S3-Lebenszyklus-Konfigurationen. Jedes Beispiel zeigt, wie Sie in jedem der Beispielszenarien das XML spezifizieren können.

Themen

- Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht
- Beispiel 6: Spezifikation einer Lebenszyklus-Konfigurationsregel für einen Bucket mit Versioning
- Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte
- Beispiel 8: Lebenszyklus-Konfigurationsregel für das Abbrechen mehrteiliger Uploads
- Beispiel 9: Lebenszykluskonfiguration mit größenbasierten Regeln

Beispiel 1: Festlegen eines Filters

Jede S3-Lebenszyklusregel enthält einen Filter, mit dem Sie eine Untermenge der Objekte in Ihrem Bucket identifizieren können, auf die sich die S3-Lebenszyklusregel bezieht. Das folgenden S3 Lifecycle-Konfigurationen zeigen Beispiele dafür, wie Sie einen Filter spezifizieren können.

- In dieser S3-Lebenszyklus-Konfigurationregel spezifiziert der Filter ein Schlüsselpräfix (`tax/`). Aus diesem Grund gilt die Regel für Objekte mit dem Schlüsselnamenpräfix `tax/`, wie beispielsweise `tax/doc1.txt` und `tax/doc2.txt`.

  Die Regel spezifiziert zwei Aktionen, die Amazon S3 zu Folgendem anweisen:

  - Übergang von Objekten in die Speicherklasse S3 Glacier Flexible Retrieval 365 Tage (ein Jahr) nach der Erstellung.
  - Objekte 3.650 Tage (10 Jahre) nach der Erstellung löschen (die `Expiration`-Aktion).

```
<LifecycleConfiguration>
  <Rule>
    <ID>Transition and Expiration Rule</ID>
    <Filter>
       <Prefix>tax/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>365</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
    <Expiration>
      <Days>3650</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

Statt das Objektalter in Tagen nach der Erstellung zu spezifizieren, können Sie für jede Aktion ein Datum festlegen. Sie können `Date` und `Days` nicht in derselben Regel kombinieren.

- Wenn Sie wollen, dass die S3-Lebenszyklusregel für alle Objekte im Bucket gilt, geben Sie ein leeres Präfix an. In der folgenden Konfiguration gibt die Regel eine `Transition`-Aktion an, die Amazon S3 anweist, Objekte 0 Tage nach der Erstellung in die S3 Glacier Flexible Retrieval-Speicherklasse zu überführen. Diese Regel bedeutet, dass die Objekte nach der Erstellung um Mitternacht UTC für die Archivierung in S3 Glacier Flexible Retrieval berechtigt sind. Weitere Informationen zu Lebenszykluseinschränkungen finden Sie unter Beschränkungen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Archive all object same-day upon creation</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>GLACIER/StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

- Sie können null oder mehrere Schlüsselnamenpräfixe und null oder mehr Objekt-Markierungen in einem Filter angeben. Der folgende Beispiel-Code wendet die S3-Lebenszyklusregel auf eine Untermenge von Objekten mit dem Schüsselpräfix `tax/` an, ebenso wie auf Objekte mit zwei Markierungen mit spezifischem Schüssel und Wert. Wenn Sie mehr als einen Filter angeben, müssen Sie das <And>-Element wie gezeigt einschließen (Amazon S3 wendet ein logisches `AND` an, um die angegebenen Filterbedingungen zu kombinieren).

```
...
<Filter>
   <And>
      <Prefix>tax/</Prefix>
      <Tag>
         <Key>key1</Key>
         <Value>value1</Value>
      </Tag>
      <Tag>
         <Key>key2</Key>
```

```
         <Value>value2</Value>
      </Tag>
    </And>
</Filter>
...
```

- Sie können Objekte nur auf Markierungen basierend filtern. Die folgende S3-Lebenszyklusregel beispielsweise wird auf Objekte angewendet, die die beiden spezifizierten Markierungen aufweisen (es wird kein Präfix angegeben).

```
...
<Filter>
   <And>
      <Tag>
         <Key>key1</Key>
         <Value>value1</Value>
      </Tag>
      <Tag>
         <Key>key2</Key>
         <Value>value2</Value>
      </Tag>
    </And>
</Filter>
...
```

> **Important**
>
> Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere S3-Lebenszyklus-Aktionen auszuführen sind. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:
>
> - Das permanente Löschen hat Vorrang vor einem Übergang.
> - Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
> - Wenn ein Objekt sowohl für einen Übergang von S3 Glacier Flexible Retrieval als auch von S3 Standard-IA (oder S3 One Zone-IA) in Frage kommt, wählt Amazon S3 den Übergang von S3 Glacier Flexible Retrieval.

Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht.

Beispiel 2: Deaktivieren einer Lebenszyklusregel

Sie können eine S3-Lebenszyklusregel vorübergehend deaktivieren. Die folgende S3-Lebenszyklus-Konfiguration spezifiziert zwei Regeln:

- Regel 1 weist Amazon S3 an, Objekte mit dem Präfix `logs/` bald nach der Erstellung in die Speicherklasse S3 Glacier Flexible Retrieval zu übertragen.
- Regel 2 weist Amazon S3 an, Objekte mit dem Präfix `documents/` bald nach der Erstellung in die Speicherklasse S3 Glacier Flexible Retrieval zu übertragen.

In der Konfiguration ist Regel 1 aktiviert und Regel 2 ist deaktiviert. Amazon S3 ignoriert deaktivierte Regeln.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule1</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule2</ID>
    <Filter>
      <Prefix>documents/</Prefix>
    </Filter>
    <Status>Disabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
```

```
  </Rule>
</LifecycleConfiguration>
```

Beispiel 3: Schichtweise Reduzierung der Speicherklasse über die Lebensdauer des Objekts

In diesem Beispiel nutzen Sie die S3-Lebenszyklus-Konfiguration, um die Speicherklasse von Objekten über ihre Lebensdauer stufenweise zu reduzieren. Diese schichtweise Reduzierung kann dazu beitragen, Speicherkosten zu reduzieren. Weitere Informationen zu Preisen finden Sie unter Amazon-S3-Preise.

Die folgende S3-Lebenszyklus-Konfiguration spezifiziert eine Regel, die auf Objekte mit Schlüsselnamenpräfix `logs/` angewendet wird. Die Regel definiert die folgenden Aktionen:

- Zwei Übergangsaktionen:
  - Übergang von Objekten in die Speicherklasse S3 Standard-IA 30 Tage nach der Erstellung.
  - Übergang von Objekten in die Speicherklasse S3 Glacier Flexible Retrieval 90 Tage nach der Erstellung.
- Eine Ablaufaktion, die Amazon S3 anweist, Objekte ein Jahr nach ihrer Erstellung zu löschen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>example-id</ID>
    <Filter>
       <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>30</Days>
      <StorageClass>STANDARD_IA</StorageClass>
    </Transition>
    <Transition>
      <Days>90</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
```

```
</LifecycleConfiguration>
```

> **Note**
>
> Sie können eine Regel verwenden, um alle S3-Lebenszyklus-Aktionen zu beschreiben, die für dieselbe Objektmenge angewendet werden (identifiziert durch den Filter). Andernfalls können Sie mehrere Regeln hinzufügen, die jeweils einen unterschiedlichen Filter angeben.

> **Important**
>
> Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere S3-Lebenszyklus-Aktionen auszuführen sind. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:
>
> - Das permanente Löschen hat Vorrang vor einem Übergang.
> - Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
> - Wenn ein Objekt sowohl für einen Übergang von S3 Glacier Flexible Retrieval als auch von S3 Standard-IA (oder S3 One Zone-IA) in Frage kommt, wählt Amazon S3 den Übergang von S3 Glacier Flexible Retrieval.
>
> Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht.

Beispiel 4: Festlegen mehrerer Regeln

Sie können mehrere Regeln angeben, wenn Sie unterschiedliche S3-Lebenszyklus-Aktionen auf unterschiedliche Objekte anwenden wollen. Die folgende S3-Lebenszyklus-Konfiguration spezifiziert zwei Regeln:

- Regel 1 gilt für Objekte mit dem Schlüsselnamenpräfix `classA/`. Sie weist Amazon S3 an, Objekte ein Jahr nach der Erstellung in die Speicherklasse S3 Glacier Flexible Retrieval zu übertragen, und diese Objekte 10 Jahre nach dem Erstellen ablaufen zu lassen.

- Regel 2 gilt für Objekte mit dem Schlüsselnamenpräfix `classB/`. Sie weist Amazon S3 an, Objekte 90 Tage nach der Erstellung in die Speicherklasse S3 Standard-IA zu übertragen, und sie ein Jahr nach dem zu löschen.

```
<LifecycleConfiguration>
    <Rule>
        <ID>ClassADocRule</ID>
        <Filter>
           <Prefix>classA/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>365</Days>
           <StorageClass>GLACIER</StorageClass>
        </Transition>
        <Expiration>
             <Days>3650</Days>
        </Expiration>
    </Rule>
    <Rule>
        <ID>ClassBDocRule</ID>
        <Filter>
            <Prefix>classB/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>90</Days>
           <StorageClass>STANDARD_IA</StorageClass>
        </Transition>
        <Expiration>
             <Days>365</Days>
        </Expiration>
    </Rule>
</LifecycleConfiguration>
```

Important

Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere S3-Lebenszyklus-Aktionen auszuführen sind. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:

- Das permanente Löschen hat Vorrang vor einem Übergang.
- Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
- Wenn ein Objekt sowohl für einen Übergang von S3 Glacier Flexible Retrieval als auch von S3 Standard-IA (oder S3 One Zone-IA) in Frage kommt, wählt Amazon S3 den Übergang von S3 Glacier Flexible Retrieval.

Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht.

Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht

Sie könnten eine S3-Lebenszyklus-Konfiguration angeben, in der Sie überlappende Präfixe oder Aktionen spezifizieren.

Im Allgemeinen gibt S3-Lebenszyklus Kostenoptimierung Vorrang. Wenn sich z. B. zwei Ablaufrichtlinien überschneiden, wird die Ablaufrichtlinie mit der kürzeren Frist durchgesetzt, sodass die Daten nicht länger als erwartet gespeichert werden. Wenn sich zwei Übergangsrichtlinien überschneiden, überführt S3 Lifecycle Ihre Objekte in die Speicherklasse mit geringeren Kosten.

In beiden Fällen versucht S3 Lifecycle den für Sie kostengünstigsten Pfad auszuwählen. Die Speicherklasse S3 Intelligent-Tiering ist von dieser Regel ausgenommen. S3 Intelligent-Tiering wird von S3 Lifecycle gegenüber jeder Speicherklasse bevorzugt, abgesehen von den Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive.

Die folgenden Beispiele zeigen, wie Amazon S3 potenzielle Konflikte löst.

Example 1: Überlappende Präfixe (kein Konflikt)

Die folgende Beispielkonfiguration weist zwei Regeln auf, die überlappenden Präfixe spezifizieren, wie folgt:

- Die erste Regel spezifiziert einen leeren Filter, d. h. alle Objekte in dem Bucket werden angesprochen.
- Die zweite Regel spezifiziert ein Schlüsselnamenpräfix (`logs/`), d. h. nur eine Untermenge von Objekten.

Regel 1 fordert Amazon S3 auf, alle Objekte ein Jahr nach der Erstellung zu löschen. Regel 2 fordert Amazon S3 auf, 30 Tage nach der Erstellung eine Teilmenge von Objekten in die S3-Standard-IA-Speicherklasse zu überführen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>30</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

Da in diesem Fall kein Konflikt besteht, überführt Amazon S3 die Objekte mit dem Präfix `logs/` 30 Tage nach der Erstellung in die Speicherklasse S3 Standard-IA. Wenn ein Objekt ein Jahr nach der Erstellung erreicht, wird es gelöscht.

Example 2: Widersprüchliche Lebenszyklus-Aktionen

In dieser Beispielkonfiguration gibt es zwei Regeln, die Amazon S3 anweisen, zwei unterschiedliche Aktionen für dieselbe Objektmenge zur selben Zeit in der Lebensdauer des Objekts auszuführen:

- Beide Regeln geben dasselbe Schlüsselnamenpräfix an, deshalb gelten beide Regeln für dieselbe Objektmenge.
- Beide Regeln spezifizieren dieselben 365 Tage nach der Erstellung des Objekts, wann die Regeln angewendet werden sollen.

- Eine Regel weist Amazon S3 an, Objekte zur S3 Standard-IA-Speicherklasse zu überführen. Eine andere Regel weist Amazon S3 an, die Objekte zur gleichen Zeit ablaufen zu lassen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>365</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

In diesem Fall wollen Sie, dass Objekte ablaufen (zu entfernen), deshalb macht es keinen Sinn, die Speicherklasse zu ändern, und Amazon S3 wählt einfach die Ablaufaktion für diese Objekte aus.

Example 3: Überlappende Präfixe, die zu widersprüchlichen Lebenszyklus-Aktionen führen

In diesem Beispiel besitzt die Konfiguration zwei Regeln, die überlappende Präfixe angeben, wie folgt:

- Regel 1 legt ein leeres Präfix fest (was für alle Objekte gilt).
- Regel 2 spezifiziert ein Schlüsselnamenpräfix (`logs/`), das eine Untermenge aller Objekte angibt.

Für die Untermenge der Objekte mit dem Schlüsselnamenpräfix `logs/` werden die S3-Lebenszyklus-Aktionen aus beiden Regeln angewendet. Eine Regel weist Amazon S3 an, Objekte 10 Tage nach

der Erstellung zu übertragen, und eine andere Regel weist Amazon S3 an, Objekte 365 Tage nach dem Erstellen zu übertragen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>10</Days>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>365</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

In diesem Fall entscheidet Amazon S3 den Übergang 10 Tage nach der Erstellung auszuführen.

Example 4: Tag-basiertes Filtern, das zu widersprüchlichen Lebenszyklus-Aktionen führt

Angenommen, Sie haben die folgende S3-Lebenszykluskonfiguration, die zwei Regeln enthält, die beide einen Tag-Filter spezifizieren:

- Regel 1 spezifiziert einen Tag-basierten Filter (`tag1/value1`). Diese Regel weist Amazon S3 an, Objekte 365 Tage nach der Erstellung in die Speicherklasse S3 Glacier Flexible Retrieval zu übertragen.
- Regel 2 spezifiziert einen Tag-basierten Filter (`tag2/value2`). Diese Regel weist Amazon S3 an, Objekte 14 Tage nach der Erstellung ablaufen zu lassen.

Die S3-Lebenszyklus-Konfiguration wird im Folgenden angezeigt.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Tag>
         <Key>tag1</Key>
         <Value>value1</Value>
      </Tag>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>GLACIER<StorageClass>
      <Days>365</Days>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Tag>
         <Key>tag2</Key>
         <Value>value2</Value>
      </Tag>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>14</Days>
    </Expiration>
   </Rule>
</LifecycleConfiguration>
```

Wenn ein Objekt beide Tags hat, muss Amazon S3 entscheiden, welche Regel befolgt werden soll. In diesem Fall lässt Amazon S3 das Objekt 14 Tage nach der Erstellung ablaufen. Das Objekt wird entfernt und die Übergangsaktion wird daher nicht angewendet.

> **Important**
>
> Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere S3-Lebenszyklus-Aktionen auszuführen sind. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:

- Das permanente Löschen hat Vorrang vor einem Übergang.
- Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
- Wenn ein Objekt sowohl für einen Übergang von S3 Glacier Flexible Retrieval als auch von S3 Standard-IA (oder S3 One Zone-IA) in Frage kommt, wählt Amazon S3 den Übergang von S3 Glacier Flexible Retrieval.

Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklus-Aktionen, und was Amazon S3 mit nichtversionierten Buckets macht.

Beispiel 6: Spezifikation einer Lebenszyklus-Konfigurationsregel für einen Bucket mit Versioning

Angenommen, Sie haben einen Versioning-fähigen Bucket, d. h. Sie haben für jedes Objekt eine aktuelle Version und keine oder mehr nicht aktuelle Versionen. (Weitere Informationen über das S3-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.) In diesem Beispiel möchten Sie den Verlauf eines Jahres verwalten und die nicht aktuellen Versionen löschen. S3-Lebenszyklus-Konfigurationen unterstützen die Beibehaltung von 1 bis 100 Versionen eines beliebigen Objekts.

Um Speicherkosten zu sparen, sollten Sie nicht aktuelle Versionen 30 Tage, nachdem sie nicht mehr aktuell werden, auf S3 Glacier Flexible Retrieval verschieben (vorausgesetzt, diese nicht aktuellen Objekte sind kalte Daten, für die Sie keinen Echtzeitzugriff benötigen). Darüber hinaus erwarten Sie, dass der häufige Zugriff auf die aktuellen Versionen 90 Tage nach der Erstellung abläuft, Sie könnten entscheiden, diese Objekte in die Speicherklasse S3 Standard-IA überzuführen.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
        <Filter>
           <Prefix></Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>90</Days>
           <StorageClass>STANDARD_IA</StorageClass>
        </Transition>
        <NoncurrentVersionTransition>
            <NoncurrentDays>30</NoncurrentDays>
```

```
            <StorageClass>GLACIER</StorageClass>
        </NoncurrentVersionTransition>
        <NoncurrentVersionExpiration>
            <NewerNoncurrentVersions>5</NewerNoncurrentVersions>
            <NoncurrentDays>365</NoncurrentDays>
        </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte

Ein Bucket mit Versioning enthält eine aktuelle Version und null oder mehr nicht aktuelle Versionen für jedes Objekt. Beachten Sie beim Löschen eines Objekts Folgendes:

- Wenn Sie keine Versions-ID in Ihrer Löschanfrage angeben, fügt Amazon S3 eine Löschmarkierung hinzu, statt das Objekt zu löschen. Das aktuelle Version wird nicht aktuell, und die Löschmarkierung wird zur aktuellen Version.
- Wenn Sie eine Versions-ID in Ihrer Löschanfrage angeben, löscht Amazon S3 die Objektversion permanent (es wird keine Löschmarkierung erstellt).
- Eine Löschmarkierung mit null nicht aktuellen Versionen wird als Löschmarkierung für das abgelaufene Objekt bezeichnet.

Dieses Beispiel zeigt ein Szenario, das Löschmarkierungen für abgelaufene Objekte in Ihrem Bucket erstellen kann, und demonstriert, wie Sie mit einer S3-Lebenszyklus-Konfiguration Amazon S3 anweisen können, die Löschmarkierungen für abgelaufene Objekte zu löschen.

Angenommen, Sie schreiben eine S3-Lebenszykluskonfiguration, die die `-NoncurrentVersionExpiration`Aktion verwendet, um die nicht aktuellen Versionen 30 Tage, nachdem sie nicht aktuell geworden sind, zu entfernen und höchstens 10 nicht aktuelle Versionen beizubehalten, wie im folgenden Beispiel gezeigt.

```
<LifecycleConfiguration>
    <Rule>
        ...
        <NoncurrentVersionExpiration>
            <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
            <NoncurrentDays>30</NoncurrentDays>
        </NoncurrentVersionExpiration>
```

```
    </Rule>
</LifecycleConfiguration>
```

Die Aktion `NoncurrentVersionExpiration` gilt nicht für die aktuellen Objektversionen. Sie entfernt nur nicht aktuelle Versionen.

Für aktuelle Objektversionen haben Sie die folgenden Optionen, ihre Lebensdauer zu verwalten, abhängig davon, ob die aktuellen Objektversionen einen definieren Lebenszyklus haben:

- Aktuelle Objektversionen folgen einem gut definierten Lebenszyklus.

  In diesem Fall können Sie eine S3-Lebenszykluskonfiguration mit der Aktion `Expiration` verwenden, um Amazon S3 anzuweisen, die aktuellen Versionen zu entfernen, wie im folgenden Beispiel gezeigt.

```
<LifecycleConfiguration>
    <Rule>
        ...
        <Expiration>
           <Days>60</Days>
        </Expiration>
        <NoncurrentVersionExpiration>
            <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
            <NoncurrentDays>30</NoncurrentDays>
        </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

  In diesem Beispiel entfernt Amazon S3 aktuelle Versionen 60 Tage nach ihrer Erstellung, indem für jede der aktuellen Objektversionen eine Löschmarkierung hinzugefügt wird. Durch diesen Vorgang wird die aktuelle Version nicht mehr aktuell und die Löschmarkierung wird zur aktuellen Version. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

  > **Note**
  >
  > Sie können nicht sowohl ein `Days` als auch `ExpiredObjectDeleteMarker`-Tag für dieselbe Regel angeben. Wenn Sie das `Days`-Tag angeben, führt Amazon S3 automatisch eine `ExpiredObjectDeleteMarker`-Bereinigung durch, wenn die Löschmarkierungen alt genug sind, um die Alterskriterien zu erfüllen. Um Löschmarkierungen zu bereinigen,

sobald sie die einzige Version werden, erstellen Sie eine separate Regel nur mit dem `ExpiredObjectDeleteMarker`-Tag.

Die `NoncurrentVersionExpiration`-Aktion in derselben S3-Lebenszyklus-Konfiguration entfernt nicht aktuelle Objekte 30 Tage, nachdem sie nicht aktuell wurden. Somit werden in diesem Beispiel 90 Tage nach der Objekterstellung alle Objektversionen dauerhaft entfernt. Obwohl während dieses Vorgangs Löschmarkierungen für abgelaufene Objekte erstellt werden, erkennt und entfernt Amazon S3 die Löschmarkierungen für abgelaufene Objekte für Sie.

- Aktuelle Objektversionen folgen keinem gut definierten Lebenszyklus.

  In diesem Fall müssen Sie die Objekte möglicherweise manuell entfernen, wenn Sie sie nicht mehr brauchen, und eine Löschmarkierungen mit einer oder mehreren nicht aktuellen Versionen erstellen. Wenn Ihre S3-Lebenszyklus-Konfiguration mit der `NoncurrentVersionExpiration`-Aktion alle nicht aktuellen Versionen löscht, haben Sie jetzt Löschmarkierungen für abgelaufene Objekte.

  Speziell für dieses Szenario stellt die S3-Lebenszykluskonfiguration die `Expiration`-Aktion bereit, mit der Sie die Löschmarkierungen für abgelaufene Objekte entfernen können.

```
<LifecycleConfiguration>
    <Rule>
       <ID>Rule 1</ID>
        <Filter>
          <Prefix>logs/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Expiration>
           <ExpiredObjectDeleteMarker>true</ExpiredObjectDeleteMarker>
        </Expiration>
        <NoncurrentVersionExpiration>
            <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
            <NoncurrentDays>30</NoncurrentDays>
        </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Setzen Sie das `ExpiredObjectDeleteMarker`-Element in der `Expiration`-Aktion auf `true`, um Amazon S3 anzuweisen, Löschmarkierungen für abgelaufene Objekte zu entfernen.

> **Note**
>
> Bei Verwendung der `ExpiredObjectDeleteMarker`-S3-Lebenszyklus-Aktion kann die Regel keinen Tag-basierten Filter angeben.

Beispiel 8: Lebenszyklus-Konfigurationsregel für das Abbrechen mehrteiliger Uploads

Sie können die mehrteiligen Upload-REST-API-Operationen von Amazon S3 verwenden, um große Objekte in Teilen hochzuladen. Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Mit Hilfe der S3-Lebenszyklus-Konfiguration können Sie Amazon S3 anweisen, unvollständige mehrteilige Uploads abzubrechen (identifiziert durch das Schlüsselnamenpräfix in der Regel), die nicht innerhalb einer bestimmten Anzahl an Tagen nach der Initiierung abgeschlossen wurden. Wenn Amazon S3 einen mehrteiligen Upload abbricht, werden alle diesem mehrteiligen Upload zugeordneten Teile gelöscht. Dieser Prozess hilft, Ihre Speicherkosten zu kontrollieren, indem Sie sicherstellen, dass Sie keine unvollständigen mehrteiligen Uploads mit Teilen haben, die in Amazon S3 gespeichert sind.

> **Note**
>
> Bei Verwendung der `AbortIncompleteMultipartUpload`-S3-Lebenszyklus-Aktion kann die Regel keinen Tag-basierten Filter angeben.

Das folgende Beispiel zeigt eine S3-Lebenszyklus-Konfiguration, die eine Regel mit der Aktion `AbortIncompleteMultipartUpload` spezifiziert. Diese Aktion leitet Amazon S3 dazu, unvollständige mehrteilige Uploads sieben Tage nach der Initiierung abzubrechen.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
        <Filter>
           <Prefix>SomeKeyPrefix/</Prefix>
        </Filter>
        <Status>rule-status</Status>
```

```
        <AbortIncompleteMultipartUpload>
          <DaysAfterInitiation>7</DaysAfterInitiation>
        </AbortIncompleteMultipartUpload>
    </Rule>
</LifecycleConfiguration>
```

Beispiel 9: Lebenszykluskonfiguration mit größenbasierten Regeln

Sie können Regeln erstellen, die Objekte nur basierend auf ihrer Größe übergehen. Sie können eine Mindestgröße (`ObjectSizeGreaterThan`) oder eine Maximalgröße (`ObjectSizeLessThan`) angeben, oder Sie können einen Bereich von Objektgrößen in Bytes angeben. Wenn Sie mehr als einen Filter verwenden, z. B. ein Präfix und eine Größenregel, müssen Sie die Filter in ein <And>-Element umfassen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Transition with a prefix and based on size</ID>
    <Filter>
       <And>
          <Prefix>tax/</Prefix>
          <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
       </And>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>365</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

Wenn Sie einen Bereich mit den `ObjectSizeGreaterThan`- und `ObjectSizeLessThan`-Elementen angeben, muss die maximale Objektgröße größer als die minimale Objektgröße sein. Wenn Sie mehr als einen Filter verwenden, müssen Sie die Filter in ein <And>-Element packen. Im folgenden Beispiel wird gezeigt, wie Sie Objekte in einem Bereich zwischen 500 und 64000 Byte angeben.

```
<LifecycleConfiguration>
    <Rule>
        ...
          <And>
```

```
            <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
            <ObjectSizeLessThan>64000</ObjectSizeLessThan>
         </And>
    </Rule>
</LifecycleConfiguration>
```

Sie können auch Regeln erstellen, um nicht aktuelle Objekte, die keine Daten enthalten, ausdrücklich ablaufen zu lassen, einschließlich nicht aktueller Löschmarkierungsobjekte, die in einem Bucket mit aktivierter Versionsverwaltung erstellt wurden. Das folgende Beispiel verwendet die `NoncurrentVersionExpiration`-Aktion angibt, um nicht aktuelle Versionen 30 Tage, nachdem sie nicht mehr aktuell sind, zu entfernen und höchstens 10 nicht aktuelle Versionen beizubehalten, wie im folgenden Beispiel gezeigt. Außerdem wird das `0bjectSizeLessThan`-Element verwendet, um nur Objekte ohne Daten zu filtern.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Expire noncurrent with size less than 1 byte</ID>
    <Filter>
       <ObjectSizeLessThan>1</ObjectSizeLessThan>
    </Filter>
    <Status>Enabled</Status>
    <NoncurrentVersionExpiration>
       <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
       <NoncurrentDays>30</NoncurrentDays>
    </NoncurrentVersionExpiration>
  </Rule>
</LifecycleConfiguration>
```

Amazon S3 Inventory

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header

in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Sie können Amazon S3 Inventory verwenden, um Ihren Speicher zu verwalten. Sie können es beispielsweise für die Prüfung und Meldung des Replikations- und Verschlüsselungsstatus Ihrer Objekte für Unternehmens-, Compliance- und regulatorische Anforderungen verwenden. Außerdem können Sie Business Workflows und Big-Data-Aufgaben mithilfe von Amazon S3 Inventory vereinfachen und beschleunigen. Es bietet Ihnen eine geplante Alternative zu den synchronen `List`-API-Operationen von Amazon S3. Amazon S3 Inventory verwendet die `List`-API-Operationen nicht zur Überprüfung Ihrer Objekte und hat keinen Einfluss auf die Anforderungsrate Ihres Buckets.

Amazon S3 Inventory stellt Ausgabedateien als Comma Separated Values (CSV), Apache-Optimized Row Columnar (ORC) oder Apache Parquet bereit, die Ihre Objekte und die zugehörigen Metadaten auf täglicher oder wöchentlicher Basis für einen S3-Bucket oder Objekte mit einem gemeinsam verwendeten Präfix (das heißt, Objekte, deren Namen mit einer allgemeinen Zeichenfolge beginnen) auflisten. Wenn Sie eine wöchentliche Bestandsabfrage einrichten, wird nach dem ersten Bericht jeden Sonntag (UTC-Zeitzone) ein Bericht generiert. Informationen zu den Preisen für Amazon S3 Inventory finden Sie unter Amazon-S3-Preise.

Sie können mehrere Bestandslisten für einen Bucket konfigurieren. Wenn Sie eine Bestandsliste konfigurieren, können Sie Folgendes angeben:

- welche Objektmetadaten in das Inventar aufgenommen werden sollen
- ob alle Objektversionen oder nur aktuelle Versionen aufgelistet werden sollen
- wo die Ausgabe der Bestandslistendatei gespeichert werden soll
- ob der Bestandsbericht täglich oder wöchentlich generiert werden soll
- ob die Bestandslistendatei verschlüsselt werden soll

Sie können Amazon S3 Inventory mit Standard-SQL-Abfragen abfragen, indem Sie Amazon Athena, Amazon Redshift Spectrum und andere Tools wie Presto, Apache Hive und Apache Spark verwenden. Weitere Informationen zur Verwendung von Athena zum Abfragen Ihrer Bestandsdaten finden Sie unter the section called “Bestandsabfrage mit Athena”.

Quell- und Ziel-Buckets

Der Bucket, dessen Objekte die Bestandserfassung auflistet, wird als Quell-Bucket bezeichnet. Der Bucket, in dem die Datei mit der Bestandsliste gespeichert ist, wird als Ziel-Bucket bezeichnet.

Quell-Bucket

Der Bestand listet die Objekte auf, die im Quell-Bucket gespeichert sind. Sie können eine Bestandsliste für einen ganzen Bucket erhalten oder die Liste nach dem Präfix eines Objektschlüsselnamens filtern.

Der Quell-Bucket:

- Enthält die Objekte, die im Bestand aufgelistet sind.
- Enthält die Konfiguration für den Bestand.

Ziel-Bucket

Amazon S3 Inventory listet Dateien auf, die in den Ziel-Bucket geschrieben werden. Um alle Bestandslisten-Dateien an einem gemeinsamen Speicherort im Ziel-Bucket zu gruppieren, können Sie ein Zielpräfix in der Bestands-Konfiguration angeben.

Der Ziel-Bucket:

- Enthält die Dateilisten für den Bestand.
- Enthält die Manifestdateien, die alle Bestandslistendateien enthalten, die im Ziel-Bucket gespeichert sind. Weitere Informationen finden Sie unter Bestandsmanifest.
- Erfordert eine Bucket-Richtlinie, um Amazon S3 die Berechtigung zu erteilen, die Eigentümerschaft an dem Bucket zu überprüfen, ebenso wie die Berechtigung, Dateien in den Bucket zu schreiben.
- Muss sich in derselben AWS-Region wie der Quell-Bucket befinden.
- Kann gleich dem Quell-Bucket sein.
- Kann einem anderen gehören AWS-Konto als dem Konto, dem der Quell-Bucket gehört.

Amazon-S3-Inventory-Liste

Eine Datei mit einer Bestandsliste enthält eine Liste der Objekte im Quell-Bucket sowie die Metadaten für jedes Objekt. Eine Bestandslistendatei wird im Ziel-Bucket in einem der folgenden Formate gespeichert:

- Als mit GZIP komprimierte CSV-Datei
- Als mit ZLIB komprimierte Apache-ORC-Datei (Optimized Row Columnar)
- Als mit Snappy komprimierte Apache-Parquet-Datei

> **Note**
>
> Es kann nicht garantiert werden, dass Objekte in Berichten von Amazon S3 Inventory auf bestimmte Weise sortiert werden.

Eine Datei mit einer Bestandsliste enthält eine Liste der Objekte im Quell-Bucket sowie die Metadaten für jedes aufgelistete Objekt:

- Bucket name (Bucket-Name) – Der Name des Buckets, für den der Bestand gilt.
- Key name (Schlüsselname) – Name des Objektschlüssels (oder Schlüssel), der das Objekt in dem Bucket eindeutig identifiziert. Bei Verwendung des CSV-Dateiformats ist der Schlüsselname URL-kodiert und muss dekodiert werden, bevor Sie ihn verwenden können.
- Version ID (Versions-ID) – ID der Objektversion. Wenn Sie das Versioning für einen Bucket aktivieren, weist Amazon S3 allen Objekten, die dem Bucket hinzugefügt werden, eine Versionsnummer zu. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets. (Dieses Feld ist nicht enthalten, wenn die Liste nur für die aktuelle Version der Objekte konfiguriert ist.)
- IsLatest – Wird auf gesetzt`True`, wenn das Objekt die aktuelle Version des Objekts ist. (Dieses Feld ist nicht enthalten, wenn die Liste nur für die aktuelle Version der Objekte konfiguriert ist.)
- Delete marker (Löschmarkierung) – Auf `True` gesetzt, wenn es sich bei dem Objekt um eine Löschmarkierung handelt. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets. (Dieses Feld wird dem Bericht automatisch hinzugefügt, wenn Sie den Bericht so konfiguriert haben, dass alle Versionen der Objekte aufgenommen werden.)
- Größe – Die Objektgröße in Byte, ohne die Größe von unvollständigen mehrteiligen Uploads, Objektmetadaten und Löschmarkierungen.
- Last modified date (Letztes Änderungsdatum) – Datum der Erstellung oder der letzten Änderung des Objekts, je nachdem, welches neuer ist.
- ETag – Der Entitäts-Tag (ETag) ist ein Hashwert des Objekts. Das ETag gibt nur Änderungen am Inhalt eines Objekts wieder, nicht an seinen Metadaten. Das ETag kann ein MD5-Digest der

Objektdaten sein, muss aber nicht. Dies hängt davon ab, wie das Objekt erstellt und verschlüsselt wurde.

- Storage class (Speicherklasse) – Die für die Speicherung des Objekts verwendete Speicherklasse. Auf `STANDARD`, `REDUCED_REDUNDANCY`, `STANDARD_IA`, `ONEZONE_IA`, `INTELLIGENT_TIERING`, `GLACIER`, `DEEP_ARCHIVE`, `OUTPOSTS`, `GLACIER_IR` oder `SNOW` festgelegt. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.
- Multipart upload flag (Markierung für mehrteiligen Upload) – Auf `True` gesetzt, wenn das Objekt als mehrteiliger Upload hochgeladen wurde. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.
- Replikationsstatus – Auf `PENDING`, `COMPLETED`, `FAILED`, oder `REPLICA` gesetzt. Weitere Informationen finden Sie unter Abrufen von Replikationsstatusinformationen.
- Verschlüsselungsstatus – Der serverseitige Verschlüsselungsstatus, je nachdem, welche Art von Verschlüsselungsschlüssel verwendet wird – ein von Amazon S3 verwalteter (SSE-S3)-Schlüssel, ein AWS Key Management Service (AWS KMS)-Schlüssel (SSE-KMS) oder ein vom Kunden bereitgestellter Schlüssel (SSE-C). Setzen Sie den Wert auf `SSE-S3`, `SSE-C`, `SSE-KMS` oder `NOT-SSE`. Der Status `NOT-SSE` bedeutet, dass das Objekt nicht mit serverseitiger Verschlüsselung verschlüsselt ist. Weitere Informationen finden Sie unter Datenschutz durch Verschlüsselung.
- S3 Object Lock retain until date (Beibehaltungsfrist für S3-Objektsperre) – Das Datum, bis zu dem das gesperrte Objekt nicht gelöscht werden kann. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.
- S3 Object Lock retention mode (Aufbewahrungsmodus für S3-Objekt) – Auf `Governance` oder `Compliance` gesetzt für Objekte, die gesperrt sind. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.
- S3 Object Lock legal hold status (Rechtlicher Aufbewahrungsstatus der S3-Objektsperre) – Auf `On` gesetzt, wenn für ein Objekt eine rechtliche Aufbewahrungsfrist gilt. Andernfalls lautet der Wert `Off`. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.
- S3 Intelligent-Tiering access tier (S3-Intelligent-Tiering-Zugriffsebene) – Zugriffsebene (häufig oder selten) des Objekts bei Speicherung in der Speicherklasse S3 Intelligent-Tiering. Setzen Sie den Wert auf `FREQUENT`, `INFREQUENT`, `ARCHIVE_INSTANT_ACCESS`, `ARCHIVE` oder `DEEP_ARCHIVE`. Weitere Informationen finden Sie unter Speicherklasse zur automatischen Optimierung von Daten mit sich ändernden oder unbekannten Zugriffsmustern.
- S3 Buckt Key status (Status des S3-Bucket-Schlüssels) – Auf `ENABLED` oder `DISABLED` gesetzt. Gibt an, ob das Objekt einen S3-Bucket-Schlüssel für SSE-KMS verwendet. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Bucket-Schlüssel.

- Prüfsummen-Algorithmus – Gibt den Algorithmus an, mit dem die Prüfsumme für das Objekt erstellt wurde.
- Objektzugriffskontrollliste – Eine Zugriffskontrollliste (ACL) für jedes Objekt, das definiert, welche AWS-Konten oder Gruppen Zugriff auf dieses Objekt erhalten und welche Art von Zugriff gewährt wird. Das Feld „Objekt-ACL“ ist im JSON-Format definiert. Ein S3-Inventarbericht enthält ACLs, die Objekten in Ihrem Quell-Bucket zugeordnet sind, auch wenn ACLs für den Bucket deaktiviert sind. Weitere Informationen finden Sie unter Arbeiten mit dem Feld „Objekt-ACL“ und Zugriffskontrolllisten (ACL) – Übersicht.

  > **Note**
  >
  > Das Feld „Objekt-ACL“ ist im JSON-Format definiert. In einem Bestandsbericht wird der Wert für das Feld „Objekt-ACL“ als Base64-codierte Zeichenfolge angezeigt. Angenommen, Sie haben das folgende Feld „Objekt-ACL“ im JSON-Format:
  >
  > ```
  > {
  >         "version": "2022-11-10",
  >         "status": "AVAILABLE",
  >         "grants": [{
  >             "canonicalId": "example-canonical-user-ID",
  >             "type": "CanonicalUser",
  >             "permission": "READ"
  >         }]
  > }
  > ```
  >
  > Das Feld „Objekt-ACL“ ist codiert und wird als die folgende Base64-codierte Zeichenfolge angezeigt:
  >
  > ```
  > eyJ2ZXJzaW9uIjoiMjAyMi0xMS0xMCIsInN0YXR1cyI6IkFWQUlMQUJMRSIsImdyYW50cyI6W3siY2Fub25pY2Fs
  > ```
  >
  > Um für das Feld „Objekt-ACL“ den dekodierten Wert in JSON abzurufen, können Sie dieses Feld in Amazon Athena abfragen. Abfragebeispiele finden Sie unter Abfragen von Amazon S3 Inventory mit Amazon Athena.

- Object owner (Besitzer des Objekts) – Der Besitzer des Objekts.

> **Note**
>
> Wenn ein Objekt basierend auf seiner Lebenszykluskonfiguration das Ende seiner Lebensdauer erreicht hat, stellt Amazon S3 das Objekt zum Entfernen in eine Warteschlange und entfernt es asynchron. Daher kann es eine Verzögerung zwischen dem Ablaufdatum und dem Datum geben, an dem Amazon S3 ein Objekt entfernt. Der Bestandsbericht enthält die Objekte, die abgelaufen sind, aber noch nicht entfernt wurden. Weitere Informationen über die Ablaufaktionen im S3-Lebenszyklus finden Sie unter Auslaufende Objekte.

Wir empfehlen, eine Lebenszyklusrichtlinie einzurichten, die alte Bestandslisten löscht. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Die `s3:PutInventoryConfiguration`-Berechtigung ermöglicht es einem Benutzer, beim Konfigurieren einer Bestandsliste sowohl alle Metadatenfelder auszuwählen, die zuvor für jedes Objekt aufgelistet wurden, als auch den Ziel-Bucket zum Speichern des Bestands anzugeben. Ein Benutzer mit Lesezugriff auf Objekte im Ziel-Bucket kann auf alle Objektmetadatenfelder zugreifen, die in der Bestandsliste verfügbar sind. Informationen zum Einschränken des Zugriffs auf Bestandsberichte finden Sie unter Gewähren von Berechtigungen für S3 Inventory und S3 Analytics.

Bestandskonsistenz

Möglicherweise erscheinen nicht alle Ihre Objekte in jeder Bestandsliste. Die Bestandsliste bietet letztendliche Datenkonsistenz für PUT-Anforderungen (sowohl neuer als auch überschriebener Objekte) sowie für `DELETE`-Anforderungen. Jede Bestandsliste für einen Bucket ist ein Snapshot der Bucket-Elemente. Diese Listen sind letztendlich konsistent (das heißt, eine Liste enthält möglicherweise keine kürzlich hinzugefügten oder gelöschten Objekte).

Um den Status eines Objekts zu überprüfen, bevor Sie eine Aktion dafür ausführen, empfehlen wir, eine `HeadObject`-REST-API-Anfrage zu stellen, um Metadaten für das Objekt abzurufen, oder die Objekteigenschaften in der Amazon S3-Konsole zu überprüfen. Sie können Objektmetadaten auch mit der AWS CLI oder den AWS -SDKs überprüfen. Weitere Informationen finden Sie unter `HeadObject` in der API-Referenz zu Amazon Simple Storage Service.

Weitere Informationen zur Arbeit mit Amazon S3 Inventory finden Sie in den folgenden Themen.

Themen

- Konfigurieren von Amazon S3 Inventory

- Einrichten von Amazon-S3-Ereignis-Benachrichtigungen für den Bestandsabschluss
- Lokalisieren Ihrer Bestandsliste
- Abfragen von Amazon S3 Inventory mit Amazon Athena
- Konvertieren leerer Versions-ID-Strings in Amazon-S3-Inventory-Berichten in Null-Zeichenfolgen
- Arbeiten mit dem Feld „Objekt-ACL“

Konfigurieren von Amazon S3 Inventory

Amazon S3 Inventory stellt eine Flat-File-Liste Ihrer Objekte und Metadaten nach einem von Ihnen definierten Zeitplan bereit. Sie können S3 Inventory als geplante Alternative zur synchronen API-Operation `List` von Amazon S3 verwenden. S3 Inventory stellt Ausgabedateien im Format Comma-Separated Values (CSV), Apache-Optimized Row Columnar (ORC) oder Apache Parquet (Parquet) bereit, in denen Ihre Objekte und die entsprechenden Metadaten aufgeführt werden.

Sie können S3 Inventory so konfigurieren, dass täglich oder wöchentlich Bestandslisten für einen S3-Bucket oder für Objekte mit gemeinsamem Präfix (Objekte, deren Namen mit derselben Zeichenfolge beginnen) erstellt werden. Weitere Informationen finden Sie unter Amazon S3 Inventory.

In diesem Abschnitt wird beschrieben, wie Sie einen Bestand, einschließlich Details über die Quell- und Ziel-Buckets des Bestands, einrichten.

Themen

- Übersicht
- Erstellen einer Ziel-Bucket-Richtlinie
- Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres vom Kunden verwalteten Schlüssels für die Verschlüsselung
- Konfigurieren des Bestands mit der S3-Konsole
- Verwendung der REST-API für die Arbeit mit S3 Inventory

Übersicht

Amazon S3 Inventory hilft Ihnen, Ihren Speicher zu verwalten, indem nach einem definierten Zeitplan Listen der Objekte in einem S3-Bucket erstellt werden. Sie können mehrere Bestandslisten für einen Bucket konfigurieren. Die Bestandslisten werden in CSV-, ORC,- oder Parquet-Dateien in einem Ziel-Bucket veröffentlicht.

Der einfachste Weg, einen Bestand einzurichten, ist die Verwendung der Amazon S3-Konsole, aber Sie können auch die Amazon S3-REST-API, AWS Command Line Interface (AWS CLI) oder AWS SDKs verwenden. Die Konsole führt den ersten Schritt des folgenden Verfahrens für Sie durch: Hinzufügen einer Bucket-Richtlinie zum Ziel-Bucket.

So richten Sie Amazon S3 Inventory für einen S3-Bucket ein

1. Fügen Sie eine Bucket-Richtlinie für den Ziel-Bucket hinzu.

   Sie müssen eine Bucket-Richtlinie für den Ziel-Bucket erstellen, die Amazon S3 Berechtigungen zum Schreiben von Objekten in den Bucket am definierten Speicherort erteilt. Eine Beispielrichtlinie finden Sie unter Gewähren von Berechtigungen für S3 Inventory und S3 Analytics.
2. Konfigurieren Sie eine Bestandsliste, um die Objekte in einem Quell-Bucket aufzulisten und die Liste in einem Ziel-Bucket zu veröffentlichen.

   Wenn Sie eine Bestandsliste für einen Quell-Bucket konfigurieren, geben Sie den Ziel-Bucket an, in dem die Liste gespeichert werden soll, und ob Sie möchten, dass die Liste täglich oder wöchentlich generiert werden soll. Sie können auch konfigurieren, ob alle Objektversionen oder nur aktuelle Versionen aufgelistet werden sollen und welche Objektmetadaten aufgenommen werden sollen.

   Einige Objektmetadatenfelder in S3-Inventory-Berichtskonfigurationen sind optional, was bedeutet, dass sie standardmäßig verfügbar sind, aber eingeschränkt werden können, wenn Sie einem Benutzer die `-s3:PutInventoryConfiguration`Berechtigung erteilen. Mithilfe des `s3:InventoryAccessibleOptionalFields` Bedingungsschlüssels können Sie steuern, ob Benutzer diese optionalen Metadatenfelder in ihre Berichte aufnehmen können.

   Weitere Informationen zu den optionalen Metadatenfeldern, die in S3 Inventory verfügbar sind, finden Sie unter OptionalFields in der API-Referenz zu Amazon Simple Storage Service. Weitere Informationen zum Einschränken des Zugriffs auf bestimmte optionale Metadatenfelder in einer Bestandskonfiguration finden Sie unter Steuern der Erstellung von S3-Inventory-Berichten.

   Sie können angeben, dass die Bestandslistendatei verschlüsselt werden soll, indem Sie die serverseitige Verschlüsselung mit einem von Amazon S3 verwalteten Schlüssel (SSE-S3) oder einem AWS Key Management Service (AWS KMS) vom Kunden verwalteten Schlüssel (SSE-KMS) verwenden.

> **Note**
>
> Die Von AWS verwalteter Schlüssel (`aws/s3`) wird für die SSE-KMS-Verschlüsselung mit S3 Inventory nicht unterstützt.

Weitere Informationen zu SSE-S3 und SSE-KMS finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung. Wenn Sie die SSE-KMS-Verschlüsselung verwenden möchten, siehe Schritt 3.

- Informationen zur Verwendung der Konsole zum Konfigurieren einer Bestandsliste finden Sie unter Konfigurieren des Bestands mit der S3-Konsole.
- Um die Amazon S3-API zum Konfigurieren einer Bestandsliste zu verwenden, verwenden Sie die PutBucketInventoryConfiguration REST-API-Operation oder das Äquivalent aus der AWS CLI oder den AWS SDKs .

3. Um die Bestandslisten-Datei mit SSE-KMS zu verschlüsseln, erteilen Sie Amazon S3 die Berechtigung, AWS KMS key zu verwenden.

   Sie können die Verschlüsselung für die Bestandslistendatei mithilfe der Amazon S3-Konsole, der Amazon S3-REST-API AWS CLI oder AWS SDKs konfigurieren. Unabhängig davon, welche Möglichkeit Sie wählen: Sie müssen Amazon S3 die Berechtigung gewähren, den vom Kunden verwalteten Schlüssel zum Verschlüsseln der Bestandsdatei zu verwenden. Sie erteilen Amazon S3 die Berechtigung, indem Sie die Schlüsselrichtlinie für den kundenverwalteten Schlüssel ändern, den Sie zur Verschlüsselung der Bestandsdatei verwenden möchten. Weitere Informationen finden Sie unter Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres vom Kunden verwalteten Schlüssels für die Verschlüsselung.

   Der Ziel-Bucket, in dem die Bestandslistendatei gespeichert wird, kann zu einem anderen AWS-Konto als zu dem Konto gehören, zu dem der Quell-Bucket gehört. Wenn Sie die SSE-KMS-Verschlüsselung für den kontoübergreifenden Betrieb von Amazon S3 Inventory verwenden, empfehlen wir Ihnen, bei der Konfiguration des S3-Bestands einen vollqualifizierten KMS-Schlüssel-ARN zu verwenden. Weitere Informationen finden Sie unter Verwenden der SSE-KMS-Verschlüsselung für kontoübergreifende Vorgänge und unter ServerSideEncryptionByDefault in der API-Referenz für Amazon Simple Storage Service.

Erstellen einer Ziel-Bucket-Richtlinie

Wenn Sie Ihre Bestandskonfiguration über die S3-Konsole erstellen, erstellt Amazon S3 automatisch eine Bucket-Richtlinie für den Ziel-Bucket, die ihm Amazon-S3-Schreibberechtigung gewährt. Wenn Sie Ihre Bestandskonfiguration jedoch über die AWS CLI, AWS SDKs oder die Amazon S3-REST-API erstellen, müssen Sie manuell eine Bucket-Richtlinie zum Ziel-Bucket hinzufügen. Weitere Informationen finden Sie unter Gewähren von Berechtigungen für S3 Inventory und S3 Analytics. Die S3-Inventory-Ziel-Bucket-Richtlinie ermöglicht es Amazon S3, Daten für die Bestandsberichte in den Bucket zu schreiben.

Wenn beim Erstellen der Bucket-Richtlinie ein Problem auftritt, erhalten Sie Anweisungen zur Fehlerbehebung. Wenn Sie beispielsweise einen Ziel-Bucket in einem anderen auswählen AWS-Konto und keine Berechtigungen zum Lesen und Schreiben in die Bucket-Richtlinie haben, wird eine Fehlermeldung angezeigt.

In diesem Fall muss der Eigentümer des Ziel-Buckets die Bucket-Richtlinie zum Ziel-Bucket hinzufügen. Wird die Richtlinie dem Ziel-Bucket nicht hinzugefügt, erhalten Sie keinen Bestandsbericht, da Amazon S3 nicht über die Berechtigung verfügt, in den Ziel-Bucket zu schreiben. Wenn der Quell-Bucket nicht dem Konto des aktuellen Benutzers gehört, muss die richtige Konto-ID des Quell-Bucket-Eigentümers in der Richtlinie ersetzt werden.

Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres vom Kunden verwalteten Schlüssels für die Verschlüsselung

Um Amazon S3 die Berechtigung zu erteilen, Ihren AWS Key Management Service (AWS KMS) vom Kunden verwalteten Schlüssel für die serverseitige Verschlüsselung zu verwenden, müssen Sie eine Schlüsselrichtlinie verwenden. Gehen Sie wie folgt vor, um Ihre Schlüsselrichtlinie zu aktualisieren, damit Sie einen vom Kunden verwalteten Schlüssel verwenden können.

So erteilen Sie Amazon S3 Berechtigungen zum Verschlüsseln mithilfe Ihres vom Kunden verwalteten Schlüssels

1. Melden Sie sich mit dem AWS-Konto , dem der vom Kunden verwaltete Schlüssel gehört, bei der an AWS Management Console.
2. Öffnen Sie die - AWS KMS Konsole unter https://console.aws.amazon.com/kms.
3. Um die zu ändern AWS-Region, verwenden Sie die Regionsauswahl in der oberen rechten Ecke der Seite.

4. Klicken Sie im linken Navigationsbereich auf Customer managed keys (Vom Kunden verwaltete Schlüssel).
5. Wählen Sie unter Kundenverwaltete Schlüssel den vom Kunden verwalteten Schlüssel aus, den Sie zum Verschlüsseln der Bestandsdateien verwenden möchten.
6. Wählen Sie im Abschnitt Key policy (Schlüsselrichtlinie) die Option Switch to policy view (Zur Richtlinienansicht wechseln) aus.
7. Um die Schlüsselrichtlinie zu aktualisieren, wählen Sie Edit (Bearbeiten).
8. Fügen Sie auf der Seite Schlüsselrichtlinie bearbeiten die folgenden Zeilen zu der vorhandenen Schlüsselrichtlinie hinzu. Geben Sie für *source-account-id* und *DOC-EXAMPLE-SOURCE-BUCKET* die entsprechenden Werte für Ihren Anwendungsfall an.

```
{
    "Sid": "Allow Amazon S3 use of the customer managed key",
    "Effect": "Allow",
    "Principal": {
        "Service": "s3.amazonaws.com"
    },
    "Action": [
        "kms:GenerateDataKey"
    ],
    "Resource": "*",
    "Condition":{
      "StringEquals":{
         "aws:SourceAccount":"source-account-id"
     },
      "ArnLike":{
        "aws:SourceARN": "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
     }
   }
}
```

9. Wählen Sie Änderungen speichern aus.

Weitere Informationen zum Erstellen von kundenverwalteten Schlüsseln in und zum Verwenden von Schlüsselrichtlinien finden Sie unter den folgenden Links im AWS Key Management Service Benutzerhandbuch:

- Verwalten von Schlüsseln
- Schlüsselrichtlinien in AWS KMS

Konfigurieren des Bestands mit der S3-Konsole

Verwenden Sie diese Anweisungen, um den Bestand mit der S3-Konsole zu konfigurieren.

> **Note**
>
> Es könnte bis zu 48 Stunden dauern, bis Amazon S3 den ersten Inventarbericht bereitstellt.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus. Wählen Sie in der Liste Bucket den Namen des Buckets aus, für den Sie Amazon S3 Inventory konfigurieren wollen.
3. Wählen Sie den Tab Management.
4. Wählen Sie unter Inventory configurations (Bestands-Konfigurationen) die Option Create inventory configuration (Bestands-Konfiguration erstellen).
5. Geben Sie unter Name der Bestandskonfiguration einen Namen ein.
6. Gehen Sie für Umfang für den Bestand wie folgt vor:
   - Geben Sie ein optionales Präfix ein.
   - Wählen Sie aus, welche Objektversionen eingeschlossen werden sollen: Nur aktuelle Versionen oder Alle Versionen einschließen.
7. Wählen Sie unter Berichtsdetails den Speicherort des AWS-Konto aus, in dem Sie die Berichte speichern möchten: Dieses Konto oder ein anderes Konto.
8. Wählen Sie unter Destination (Ziel) den Ziel-Bucket aus, in dem die Bestandsberichte gespeichert werden sollen.

   Der Ziel-Bucket muss sich in derselben befinden AWS-Region wie der Bucket, für den Sie den Bestand einrichten. Der Ziel-Bucket kann sich in einem anderem AWS-Konto befinden. Beim Festlegen des Ziel-Buckets können Sie auch ein optionales Präfix angeben, um Ihre Bestandsberichte zu gruppieren.

   Unter dem Bucket-Feld Destination (Ziel) sehen Sie die Anweisung Ziel-Bucket-Berechtigung, die der Ziel-Bucket-Richtlinie hinzugefügt wird, damit Amazon S3 Daten in diesen Bucket platzieren kann. Weitere Informationen finden Sie unter Erstellen einer Ziel-Bucket-Richtlinie.
9. Wählen Sie unter Häufigkeit aus, wie oft der Bericht erstellt wird: Täglich oder Wöchentlich.

10. Wählen Sie als Ausgabeformat eines der folgenden Formate für den Bericht aus:

    - CSV – Wenn Sie diesen Bestandsbericht mit S3-Batchoperationen verwenden oder in einem anderen Tool wie Microsoft Excel analysieren möchten, wählen Sie CSV aus.
    - Apache ORC
    - Apache Parquet

11. Wählen Sie unter Status die Option Enable (Aktivieren) oder Disable (Deaktivieren) aus.

12. Führen Sie unter Verschlüsselung des Bestandsberichts die folgenden Schritte aus, um die serverseitige Verschlüsselung zu konfigurieren:

    a. Wählen Sie unter Serverseitige Verschlüsselung entweder Verschlüsselungsschlüssel nicht angeben oder Verschlüsselungsschlüssel zum Verschlüsseln von Daten angeben aus.

    - Um die Bucket-Einstellungen für die serverseitige Standardverschlüsselung von Objekten beizubehalten, wenn sie in Amazon S3 gespeichert werden, wählen Sie Keinen Verschlüsselungsschlüssel angeben aus. Solange S3-Bucket-Schlüssel für den Ziel-Bucket aktiviert sind, wendet der Kopiervorgang S3-Bucket-Schlüssel auf den Ziel-Bucket an.

      > **Note**
      >
      > Wenn die Bucket-Richtlinie für das angegebene Ziel vorschreibt, dass Objekte verschlüsselt werden müssen, bevor sie in Amazon S3 gespeichert werden, müssen Sie Verschlüsselungsschlüssel angeben auswählen. Andernfalls schlägt das Kopieren von Objekten in das Ziel fehl.

    - Um Objekte zu verschlüsseln, bevor sie in Amazon S3 gespeichert werden, wählen Sie Verschlüsselungsschlüssel angeben aus.

    b. Wenn Sie Verschlüsselungsschlüssel angeben ausgewählt haben, müssen Sie unter Verschlüsselungstyp entweder Von Amazon S3 verwalteter Schlüssel (SSE-S3) oder -AWS Key Management Service Schlüssel (SSE-KMS) auswählen.

    SSE-S3 verwendet für die Verschlüsselung der einzelnen Objekte eine der stärksten Blockverschlüsselungen: 256-bit Advanced Encryption Standard (AES-256). Mit SSE-KMS erhalten Sie mehr Kontrolle über Ihren Schlüssel. Weitere Informationen zu SSE-S3 finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten

Schlüsseln (SSE-S3). Weitere Informationen zu SSE-KMS finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

> **Note**
>
> Um die Bestandslisten-Datei mit SSE-KMS zu verschlüsseln, müssen Sie Amazon S3 die Berechtigung erteilen, den vom Kunden verwalteten Schlüssel zu verwenden. Anleitungen finden Sie unter Erteilen einer Amazon S3-Berechtigung zur Verschlüsselung mit Ihren KMS-Schlüsseln.

c. Wenn Sie AWS Key Management Service Schlüssel (SSE-KMS) ausgewählt haben, können AWS KMS keySie unter Ihren AWS KMS Schlüssel über eine der folgenden Optionen angeben.

> **Note**
>
> Wenn der Ziel-Bucket, in dem die Bestandslistendatei gespeichert ist, einem anderen gehört AWS-Konto, stellen Sie sicher, dass Sie einen vollqualifizierten KMS-Schlüssel-ARN verwenden, um Ihren KMS-Schlüssel anzugeben.

- Um aus einer Liste der verfügbaren KMS-Schlüssel auszuwählen, wählen Sie Aus Ihren AWS KMS Schlüsseln auswählen und wählen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung aus der Liste der verfügbaren Schlüssel aus. Stellen Sie sicher, dass sich der KMS-Schlüssel in derselben Region wie Ihr Bucket befindet.

  > **Note**
  >
  > Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in der Liste angezeigt. Die Von AWS verwalteter Schlüssel (`aws/s3`) wird jedoch für die SSE-KMS-Verschlüsselung mit S3 Inventory nicht unterstützt.

- Um den KMS-Schlüssel-ARN einzugeben, wählen Sie AWS KMS Schlüssel-ARN eingeben und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
- Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

13. Wählen Sie für Zusätzliche Metadatenfelder eine oder mehrere der folgenden Optionen für das Hinzufügen zum Bestandsbericht aus:

    - Größe – Die Objektgröße in Byte, ohne die Größe von unvollständigen mehrteiligen Uploads, Objektmetadaten und Löschmarkierungen.
    - Last modified date (Letztes Änderungsdatum) – Datum der Erstellung oder der letzten Änderung des Objekts, je nachdem, welches neuer ist.
    - Multipart upload (Mehrteiliger Upload) – Gibt an, dass das Objekt als mehrteiliger Upload hochgeladen wurde. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.
    - Replication status (Replikationsstatus) – Der Replikationsstatus des Objekts. Weitere Informationen finden Sie unter Abrufen von Replikationsstatusinformationen.
    - Encryption status (Verschlüsselungsstatus) – Der serverseitige Verschlüsselungstyp, der für die Verschlüsselung des Objekts verwendet wird. Weitere Informationen finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung.
    - Bucket-Schlüsselstatus – Gibt an, ob ein von generierter Schlüssel auf Bucket-Ebene auf das Objekt AWS KMS angewendet wird. Weitere Informationen finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln.
    - Objektzugriffskontrollliste – Eine Zugriffskontrollliste (ACL) für jedes Objekt, das definiert, welche AWS-Konten oder Gruppen Zugriff auf dieses Objekt erhalten und welche Art von Zugriff gewährt wird. Weitere Hinweise zu diesem Feld finden Sie unter Arbeiten mit dem Feld „Objekt-ACL“. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht.
    - Object owner (Besitzer des Objekts) – Der Besitzer des Objekts.
    - Storage class (Speicherklasse) – Die für die Speicherung des Objekts verwendete Speicherklasse.
    - Intelligent-Tiering: Access tier (Zugriffsebene) – Zugriffsebene (häufig oder selten) des Objekts, wenn es im S3-Intelligent-Tiering-Speicher gespeichert wurde. Weitere Informationen finden Sie unter Speicherklasse zur automatischen Optimierung von Daten mit sich ändernden oder unbekannten Zugriffsmustern.
    - ETag – Der Entitäts-Tag (ETag) ist ein Hashwert des Objekts. Das ETag gibt nur Änderungen am Inhalt eines Objekts wieder, nicht an seinen Metadaten. Das ETag kann ein MD5 Digest der Objektdaten sein, muss aber nicht. Dies hängt davon ab, wie das Objekt erstellt und verschlüsselt wurde. Weitere Informationen finden Sie unter Object in der API-Referenz zu Amazon Simple Storage Service.

- Prüfsummen-Algorithmus – Gibt den Algorithmus an, mit dem die Prüfsumme für das Objekt erstellt wurde.
- All Object lock configurations (Alle Objektsperre-Konfigurationen) – Der Objektsperrenstatus des Objekts, einschließlich der folgenden Einstellungen:
  - Object Lock: Retention mode (Aufbewahrungsmodus der Objektsperre) – Die auf das Objekt angewendete Schutzebene, entweder Governance oder Compliance.
  - Object Lock: Retain until date (Beibehaltungsfrist für Objektsperre) – Das Datum, bis zu dem das gesperrte Objekt nicht gelöscht werden kann.
  - Object Lock: Legal hold status (Status der gesetzlichen Sperrfrist des Objekts) – Der Status der gesetzlichen Sperrfrist des gesperrten Objekts.

  Weitere Informationen zur S3-Objektsperre finden Sie unter So funktioniert die S3-Objektsperre.

Weitere Informationen zum Inhalt eines Bestandsberichts finden Sie unter Amazon-S3-Inventory-Liste.

Weitere Informationen zum Einschränken des Zugriffs auf bestimmte optionale Metadatenfelder in einer Bestandskonfiguration finden Sie unter Steuern der Erstellung von S3-Inventory-Berichten.

14. Wählen Sie Erstellen.

Wenn eine Bestandsliste veröffentlicht wird, können Sie die Bestandslisten-Datei mit Amazon S3 Select abfragen. Weitere Informationen dazu, wie Sie Ihre Bestandsliste finden und die Bestandslisten-Datei mit Amazon S3 Select abfragen, finden Sie unter Lokalisieren Ihrer Bestandsliste.

Verwendung der REST-API für die Arbeit mit S3 Inventory

Im Folgenden finden Sie die REST-Operationen, die Sie für die Arbeit mit Amazon S3 Inventory verwenden können.

- DeleteBucketInventoryConfiguration
- GetBucketInventoryConfiguration
- ListBucketInventoryConfigurations
- PutBucketInventoryConfiguration

Einrichten von Amazon-S3-Ereignis-Benachrichtigungen für den Bestandsabschluss

Sie können eine Amazon S3-Ereignis-Benachrichtigung einrichten, um informiert zu werden, wenn die Prüfsummendatei für das Manifest erstellt wird, woran zu erkennen ist, dass dem Ziel-Bucket eine Bestandsliste hinzugefügt wurde. Das Manifest ist eine up-to-date Liste aller Bestandslisten am Zielspeicherort.

Amazon S3 kann Ereignisse in einem Amazon-Simple-Notification-Service-Thema (Amazon SNS), einer Amazon-Simple-Queue-Service-Warteschlange (Amazon SQS) oder einer AWS Lambda -Funktion veröffentlichen. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Die folgende Benachrichtigungskonfiguration definiert, dass alle `manifest.checksum`-Dateien, die dem Ziel-Bucket hinzugefügt werden, von der AWS Lambda `cloud-function-list-write`. verarbeitet werden.

```
<NotificationConfiguration>
  <QueueConfiguration>
      <Id>1</Id>
      <Filter>
          <S3Key>
              <FilterRule>
                  <Name>prefix</Name>
                  <Value>destination-prefix/source-bucket</Value>
              </FilterRule>
              <FilterRule>
                  <Name>suffix</Name>
                  <Value>checksum</Value>
              </FilterRule>
          </S3Key>
      </Filter>
      <Cloudcode>arn:aws:lambda:us-west-2:222233334444:cloud-function-list-write</
Cloudcode>
      <Event>s3:ObjectCreated:*</Event>
  </QueueConfiguration>
  </NotificationConfiguration>
```

Weitere Informationen finden Sie unter Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda -Entwicklerhandbuch.

Lokalisieren Ihrer Bestandsliste

Wenn eine Bestandsliste veröffentlicht wird, werden die Manifestdateien am folgenden Standort im Ziel-Bucket veröffentlicht.

```
destination-prefix/source-bucket/config-ID/YYYY-MM-DDTHH-MMZ/manifest.json
 destination-prefix/source-bucket/config-ID/YYYY-MM-DDTHH-MMZ/manifest.checksum
 destination-prefix/source-bucket/config-ID/hive/dt=YYYY-MM-DD-HH-MM/symlink.txt
```

- *destination-prefix* ist das Präfix des Objektschlüsselnamens, das optional in der Bestandskonfiguration festgelegt wird. Sie können dieses Präfix verwenden, um alle Bestandslisten-Dateien an einem gemeinsamen Standort im Ziel-Bucket zu gruppieren.
- *source-bucket* ist der Quell-Bucket, für den die Bestandsliste erstellt wird. Der Quell-Bucket-Name wird hinzugefügt, um Kollisionen zu vermeiden, wenn mehrere Bestandsberichte von mehreren Quell-Buckets an denselben Ziel-Bucket gesendet werden.
- *config-ID* wird hinzugefügt, um Kollisionen zu vermeiden, wenn mehrere Bestandsberichte vom selben Quell-Bucket an denselben Ziel-Bucket gesendet werden. Die *config-ID* kommt aus der Bestandsbericht-Konfiguration und ist der Name des Berichts, der bei der Einrichtung festgelegt wurde.
- *YYYY-MM-DDTHH-MMZ* ist der Zeitstempel, der sich aus der Startzeit und dem Datum zusammensetzt, an dem die Bestandsberichtserstellung mit dem Scannen des Buckets beginnt, z. B. `2016-11-06T21-32Z`.
- `manifest.json` ist die Manifestdatei.
- `manifest.checksum` ist das MD5-Hash des Inhalts der `manifest.json`-Datei.
- `symlink.txt` ist die mit Apache Hive kompatible Manifest-Datei.

Die Bestandslisten werden täglich oder wöchentlich am folgenden Standort im Ziel-Bucket veröffentlicht.

```
destination-prefix/source-bucket/config-ID/data/example-file-name.csv.gz
...
destination-prefix/source-bucket/config-ID/data/example-file-name-1.csv.gz
```

- *destination-prefix* ist das Präfix des Objektschlüsselnamens, das optional in der Bestandskonfiguration festgelegt wird. Es kann verwendet werden, um alle Bestandslisten-Dateien an einem gemeinsamen Standort innerhalb des Ziel-Buckets zu gruppieren.

- *source-bucket* ist der Quell-Bucket, für den die Bestandsliste erstellt wird. Der Quell-Bucket-Name wird hinzugefügt, um Kollisionen zu vermeiden, wenn mehrere Bestandsberichte von mehreren Quell-Buckets an denselben Ziel-Bucket gesendet werden.
- *example-file-name*`.csv.gz` ist eine der CSV-Bestandsdateien. ORC-Bestandsnamen enden mit der Dateinamenserweiterung `.orc` und Parquet-Bestandsnamen enden mit der Dateinamenserweiterung `.parquet`.

  Sie können eine Bestandslisten-Datei mit Amazon S3 Select abfragen. Wählen Sie in der Amazon S3-Konsole den Namen der Bestandsliste aus (z. B. *destination-prefix* /*source-bucket* /*config-ID* /data/*example-file-name.csv.gz*). Wählen Sie dann Objektaktionen und Abfragen mit S3 Select aus. Ein Beispiel für die Verwendung einer S3 Select-Aggregatfunktion zum Abfragen einer Bestandslisten-Datei finden Sie unter SUMBeispiel für .

Bestandsmanifest

Die Manifest-Dateien `manifest.json` und `symlink.txt` beschreiben, wo sich die Bestandsdateien befinden. Wenn eine neue Bestandsliste geliefert wird, wird sie durch eine neue Reihe von Manifestdateien begleitet. Diese Dateien könnten sich gegenseitig überschreiben. In versionierungsfähigen Buckets erstellt Amazon S3 neue Versionen der Manifestdateien.

Jedes in der `manifest.json`-Datei enthaltene Manifest bietet Metadaten und andere grundlegende Informationen zu einem Bestand. Diese Informationen beinhalten Folgendes:

- Name des Quell-Buckets
- Name des Ziel-Buckets
- Bestandsversion
- Erstellungszeitstempel im Epochen-Datumsformat, der aus der Startzeit und dem Datum besteht, an dem die Bestandberichtserstellung beginnt, den Bucket zu scannen
- Format und Schema der Bestandsdateien
- Liste der Bestandsdateien, die im Ziel-Bucket enthalten sind

Immer wenn eine `manifest.json`-Datei geschrieben wird, wird sie von einer `manifest.checksum`-Datei begleitet, die das MD5-Hash des Inhalts der `manifest.json`-Datei ist.

Example Bestandsmanifest in einer **`manifest.json`**-Datei

Die folgenden Beispiele zeigen ein Bestandsmanifest in einer `manifest.json`-Datei für CSV-, ORC- und Parquet-formatierte Bestände.

CSV

Nachfolgend finden Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen CSV-formatierten Bestand.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-inventory-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
    "fileFormat": "CSV",
    "fileSchema": "Bucket, Key, VersionId, IsLatest, IsDeleteMarker,
 Size, LastModifiedDate, ETag, StorageClass, IsMultipartUploaded,
 ReplicationStatus, EncryptionStatus, ObjectLockRetainUntilDate, ObjectLockMode,
 ObjectLockLegalHoldStatus, IntelligentTieringAccessTier, BucketKeyStatus,
 ChecksumAlgorithm, ObjectAccessControlList, ObjectOwner",
    "files": [
        {
            "key": "Inventory/example-source-bucket/2016-11-06T21-32Z/
files/939c6d46-85a9-4ba8-87bd-9db705a579ce.csv.gz",
            "size": 2147483647,
            "MD5checksum": "f11166069f1990abeb9c97ace9cdfabc"
        }
    ]
}
```

ORC

Nachfolgend finden Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen ORC-formatierten Bestand.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
    "fileFormat": "ORC",
```

```
    "fileSchema":
 "struct<bucket:string,key:string,version_id:string,is_latest:boolean,is_delete_marker:boole
    "files": [
        {
            "key": "inventory/example-source-bucket/data/
d794c570-95bb-4271-9128-26023c8b4900.orc",
            "size": 56291,
            "MD5checksum": "5925f4e78e1695c2d020b9f6eexample"
        }
    ]
}
```

Parquet

Nachfolgend finden Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen Parquet-formatierten Bestand.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
    "fileFormat": "Parquet",
    "fileSchema": "message s3.inventory { required binary bucket (UTF8);
 required binary key (UTF8); optional binary version_id (UTF8); optional boolean
 is_latest; optional boolean is_delete_marker; optional int64 size; optional
 int64 last_modified_date (TIMESTAMP_MILLIS); optional binary e_tag (UTF8);
 optional binary storage_class (UTF8); optional boolean is_multipart_uploaded;
 optional binary replication_status (UTF8); optional binary encryption_status
 (UTF8); optional int64 object_lock_retain_until_date (TIMESTAMP_MILLIS); optional
 binary object_lock_mode (UTF8); optional binary object_lock_legal_hold_status
 (UTF8); optional binary intelligent_tiering_access_tier (UTF8); optional binary
 bucket_key_status (UTF8); optional binary checksum_algorithm (UTF8); optional
 binary object_access_control_list (UTF8); optional binary object_owner (UTF8);}",
    "files": [
        {
           "key": "inventory/example-source-bucket/data/
d754c470-85bb-4255-9218-47023c8b4910.parquet",
            "size": 56291,
            "MD5checksum": "5825f2e18e1695c2d030b9f6eexample"
        }
    ]
}
```

Die `symlink.txt`-Datei ist eine mit Apache Hive kompatible Manifest-Datei, mit der Bestandsdateien und die zugehörigen Datendateien automatisch entdecken kann. Das Hive-kompatible Manifest funktioniert mit den Hive-kompatiblen Services Athena und Amazon Redshift Spectrum. Außerdem funktioniert dies mit Hive-kompatiblen Anwendungen, einschließlich Presto, Apache Hive, Apache Spark und vielen anderen.

> **Important**
>
> Die mit Apache Hive kompatible Manifest-Datei `symlink.txt` funktioniert derzeit nicht mit AWS Glue.
> Das Lesen der Datei `symlink.txt` mit Apache Hive und Apache Spark wird für ORC- und Parquet-formatierte Bestandsdateien nicht unterstützt.

Abfragen von Amazon S3 Inventory mit Amazon Athena

Sie können Dateien aus Amazon S3 Inventory mit Standard-SQL-Abfragen abfragen, indem Sie Amazon Athena in allen Regionen verwenden, in denen Athena verfügbar ist. Informationen zur Überprüfung der Verfügbarkeit von AWS-Region finden Sie in der AWS-Region-Tabelle.

Athena kann Amazon-S3-Inventory-Dateien in Format Apache-ORC-Spalten (Optimized Row Columnar), Apache Parquet oder CSV (durch Komma getrennte Werte) abfragen. Wenn Sie Athena für die Abfrage der Bestandsdateien verwenden, empfehlen wir, dass Sie ORC- oder Parquet-formatierte Bestandsdateien verwenden. Die Formate ORC und Parquet bieten eine schnellere Abfrageleistung und niedrigere Abfragekosten. ORC und Parquet sind selbstbeschreibende, typerkennende und spaltenbasierte Datenformate, die für Apache Hadoop entwickelt wurden. Das spaltenbasierte Format lässt den Leser nur die Spalten lesen, entpacken und verarbeiten, die für die aktuelle Abfrage benötigt werden. Die Formate ORC und Parquet für Amazon S3 Inventory sind in allen AWS-Regionen verfügbar.

So fragen Sie Amazon-S3-Inventory-Dateien mit Athena ab

1. Erstellen Sie eine Athena-Tabelle. Informationen zum Erstellen einer Tabelle finden Sie unter Erstellen von Tabellen in Amazon Athena im Amazon Athena-Benutzerhandbuch.
2. Erstellen Sie Ihre Abfrage mithilfe einer der folgenden Beispielabfragevorlagen, je nachdem, ob Sie einen ORC-formatierten, einen Parquet-formatierten oder einen CSV-formatierten Bestandsbericht abfragen.

- Bei Verwendung von Athena für die Abfrage eines ORC-formatierten Bestandsberichts verwenden Sie die folgende Beispielabfrage als Vorlage.

  Die folgende Beispielabfrage umfasst alle optionalen Felder in einem nach ORC formatierten Bestandsbericht.

  Gehen Sie wie folgt vor, um diese Beispielabfrage zu verwenden:

  - Ersetzen Sie *your_table_name* durch den Namen der Athena-Tabelle, die Sie erstellt haben.
  - Entfernen Sie alle optionalen Felder, die Sie für Ihren Bestand nicht ausgewählt haben, sodass die Abfrage den für Ihren Bestand gewählten Feldern entspricht.
  - Ersetzen Sie den folgenden Bucket-Namen und den Bestandsort (die Konfigurations-ID) entsprechend Ihrer Konfiguration.

    `s3://`*DOC-EXAMPLE-BUCKET*`/`*config-ID*`/hive/`
  - Ersetzen Sie das Datum *2022-01-01-00-00* unter `projection.dt.range` mit dem ersten Tag des Zeitbereichs, innerhalb dessen Sie die Daten in Athena partitionieren. Weitere Informationen finden Sie unter Partitionieren von Daten in Athena.

```
CREATE EXTERNAL TABLE your_table_name(
         bucket string,
         key string,
         version_id string,
         is_latest boolean,
         is_delete_marker boolean,
         size bigint,
         last_modified_date timestamp,
         e_tag string,
         storage_class string,
         is_multipart_uploaded boolean,
         replication_status string,
         encryption_status string,
         object_lock_retain_until_date bigint,
         object_lock_mode string,
         object_lock_legal_hold_status string,
         intelligent_tiering_access_tier string,
         bucket_key_status string,
         checksum_algorithm string,
         object_access_control_list string,
```

```
         object_owner string
) PARTITIONED BY (
        dt string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.orc.OrcSerde'
  STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
  OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat'
  LOCATION 's3://source-bucket/config-ID/hive/'
  TBLPROPERTIES (
    "projection.enabled" = "true",
    "projection.dt.type" = "date",
    "projection.dt.format" = "yyyy-MM-dd-HH-mm",
    "projection.dt.range" = "2022-01-01-00-00,NOW",
    "projection.dt.interval" = "1",
    "projection.dt.interval.unit" = "HOURS"
  );
```

- Wenn Sie Athena verwenden, um einen Parquet-formatierter Bestandsbericht abzufragen, verwenden Sie die Beispielabfrage für einen Bericht im ORC-Format. Verwenden Sie jedoch folgenden Parquet-SerDe anstelle des ORC-SerDe in der`ROW FORMAT SERDE`-Anweisung.

  ```
  ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
  ```

- Bei Verwendung von Athena für die Abfrage eines CSV-formatierten Bestandsberichts verwenden Sie die folgende Beispielabfrage als Vorlage.

  Die folgende Beispielabfrage umfasst alle optionalen Felder in einem nach CSV formatierten Bestandsbericht.

  Gehen Sie wie folgt vor, um diese Beispielabfrage zu verwenden:

  - Ersetzen Sie *`your_table_name`* durch den Namen der Athena-Tabelle, die Sie erstellt haben.
  - Entfernen Sie alle optionalen Felder, die Sie für Ihren Bestand nicht ausgewählt haben, sodass die Abfrage den für Ihren Bestand gewählten Feldern entspricht.
  - Ersetzen Sie den folgenden Bucket-Namen und den Bestandsort (die Konfigurations-ID) entsprechend Ihrer Konfiguration.

    `s3://`*`DOC-EXAMPLE-BUCKET`*`/`*`config-ID`*`/hive/`

- Ersetzen Sie das Datum *2022-01-01-00-00* unter `projection.dt.range` mit dem ersten Tag des Zeitbereichs, innerhalb dessen Sie die Daten in Athena partitionieren. Weitere Informationen finden Sie unter Partitionieren von Daten in Athena.

```
CREATE EXTERNAL TABLE your_table_name(
         bucket string,
         key string,
         version_id string,
         is_latest boolean,
         is_delete_marker boolean,
         size string,
         last_modified_date string,
         e_tag string,
         storage_class string,
         is_multipart_uploaded boolean,
         replication_status string,
         encryption_status string,
         object_lock_retain_until_date string,
         object_lock_mode string,
         object_lock_legal_hold_status string,
         intelligent_tiering_access_tier string,
         bucket_key_status string,
         checksum_algorithm string,
         object_access_control_list string,
         object_owner string
) PARTITIONED BY (
        dt string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
  STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
  OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat'
  LOCATION 's3://source-bucket/config-ID/hive/'
  TBLPROPERTIES (
    "projection.enabled" = "true",
    "projection.dt.type" = "date",
    "projection.dt.format" = "yyyy-MM-dd-HH-mm",
    "projection.dt.range" = "2022-01-01-00-00,NOW",
    "projection.dt.interval" = "1",
    "projection.dt.interval.unit" = "HOURS"
  );
```

3. Sie können jetzt verschiedene Bestandsabfragen ausführen, wie in den folgenden Beispielen gezeigt. Ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

```
# Get a list of the latest inventory report dates available.
SELECT DISTINCT dt FROM your_table_name ORDER BY 1 DESC limit 10;

# Get the encryption status for a provided report date.
SELECT encryption_status, count(*) FROM your_table_name WHERE dt = 'YYYY-MM-DD-HH-
MM' GROUP BY encryption_status;

# Get the encryption status for inventory report dates in the provided range.
SELECT dt, encryption_status, count(*) FROM your_table_name
WHERE dt > 'YYYY-MM-DD-HH-MM' AND dt < 'YYYY-MM-DD-HH-MM' GROUP BY dt,
 encryption_status;
```

Wenn Sie S3 Inventory so konfigurieren, dass das Feld „Objekt-Zugriffssteuerungsliste (ACL)“ zu einem Bestandsbericht hinzugefügt wird, zeigt der Bericht den Wert für das Feld „Objekt-ACL“ als Base64-codierte Zeichenfolge an. Um für das Feld „Objekt-ACL“ den dekodierten Wert in JSON abzurufen, können Sie dieses Feld mit Athena abfragen. Sehen Sie sich die folgenden Abfragebeispiele an. Weitere Informationen zum Feld „Objekt-ACL“ erhalten Sie unter Arbeiten mit dem Feld „Objekt-ACL“.

```
# Get the S3 keys that have Object ACL grants with public access.
WITH grants AS (
    SELECT key,
        CAST(
            json_extract(from_utf8(from_base64(object_access_control_list)),
 '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))
        ) AS grants_array
    FROM your_table_name
)
SELECT key,
       grants_array,
       grant
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE element_at(grant, 'uri') = 'http://acs.amazonaws.com/groups/global/AllUsers'
```

```
# Get the S3 keys that have Object ACL grantees in addition to the object owner.
WITH grants AS
    (SELECT key,
    from_utf8(from_base64(object_access_control_list)) AS
 object_access_control_list,
```

```
         object_owner,
         CAST(json_extract(from_utf8(from_base64(object_access_control_list)),
         '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))) AS grants_array
    FROM your_table_name)
SELECT key,
       grant,
       objectowner
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE cardinality(grants_array) > 1 AND element_at(grant, 'canonicalId') !=
 object_owner;
```

```
# Get the S3 keys with READ permission that is granted in the Object ACL.
WITH grants AS (
    SELECT key,
        CAST(
            json_extract(from_utf8(from_base64(object_access_control_list)),
 '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))
        ) AS grants_array
    FROM your_table_name
)
SELECT key,
       grants_array,
       grant
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE element_at(grant, 'permission') = 'READ';
```

```
# Get the S3 keys that have Object ACL grants to a specific canonical user ID.
WITH grants AS (
    SELECT key,
        CAST(
            json_extract(from_utf8(from_base64(object_access_control_list)),
 '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))
        ) AS grants_array
    FROM your_table_name
)
SELECT key,
       grants_array,
       grant
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE element_at(grant, 'canonicalId') = 'user-canonical-id';
```

```
# Get the number of grantees on the Object ACL.
SELECT key,
       object_access_control_list,
       json_array_length(json_extract(object_access_control_list,'$.grants')) AS
 grants_count
FROM your_table_name;
```

Weitere Informationen zur Verwendung von Athena finden Sie im Amazon Athena-Benutzerhandbuch.

Konvertieren leerer Versions-ID-Strings in Amazon-S3-Inventory-Berichten in Null-Zeichenfolgen

> **Note**
>
> Das folgende Verfahren gilt nur für Amazon S3-Lagerbestandsberichte, die alle Versionen enthalten, und nur, wenn die Berichte „Alle Versionen“ als Manifeste für S3-Batch-Operationen bei Buckets verwendet werden, bei denen S3-Versionierung aktiviert ist. Sie müssen keine Zeichenfolgen für S3-Inventory-Berichte konvertieren, die nur die aktuelle Version angeben.

Sie können S3-Inventory-Berichte als Manifeste für S3-Batchvorgänge verwenden. Wenn jedoch die S3-Versionsverwaltung für einen Bucket aktiviert ist, markieren S3-Inventory-Berichte, die alle Versionen enthalten, alle nullversionierten Objekte mit leeren Zeichenfolgen im Versions-ID-Feld. Wenn ein Inventory-Bericht alle Objektversions-IDs enthält, erkennen Batchvorgänge `null`-Zeichenfolgen als Versions-IDs, jedoch keine leeren Zeichenfolgen.

Wenn ein S3-Batchvorgangauftrag einen S3-Inventory-Bericht „alle Versionen“ als Manifest verwendet, schlägt er alle Aufgaben für Objekte fehl, die eine leere Zeichenfolge im Feld Versions-ID haben. Gehen Sie wie folgt vor, um leere Zeichenfolgen im Versions-ID-Feld des S3-Inventory-Berichts in `null`-Zeichenfolgen für Batchvorgänge zu konvertieren.

Aktualisieren eines Amazon-S3-Inventory-Berichts zur Verwendung mit Batchvorgängen

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Navigieren Sie zu Ihrem S3-Inventory-Bericht. Der Inventory-Bericht befindet sich im Ziel-Bucket, den Sie beim Konfigurieren Ihres Inventory-Berichts angegeben haben. Weitere Informationen zum Auffinden von Inventory-Berichten finden Sie unter Lokalisieren Ihrer Bestandsliste.

   a. Wählen Sie den Ziel-Bucket aus.

   b. Wählen Sie den Ordner aus. Der Ordner ist nach dem ursprünglichen Quell-Bucket benannt.

   c. Wählen Sie den nach der Inventory-Konfiguration benannten Ordner aus.

   d. Aktivieren Sie das Kontrollkästchen neben dem Ordner namens Hive aus. Wählen Sie oben auf der Seite S3-URI kopieren aus, um den S3-URI für den Ordner zu kopieren.
3. Öffnen Sie die Amazon-Athena-Konsole unter https://console.aws.amazon.com/athena/.
4. Wählen Sie im Abfrage-Editor Einstellungen und anschließend Verwalten aus. Wählen Sie auf der Seite Einstellungen verwalten für Standort des Abfrageergebnisses einen S3-Bucket aus, um Ihre Abfrageergebnisse zu speichern.
5. Erstellen Sie im Abfrage-Editor eine Athena-Tabelle, um die Daten mit dem folgenden Befehl im Inventory-Bericht zu speichern. Ersetzen Sie *table_name* durch einen Namen Ihrer Wahl, und fügen Sie in der LOCATION-Klausel den zuvor kopierten S3-URI ein. Wählen Sie dann Ausführen aus, um die Abfrage auszuführen.

```
CREATE EXTERNAL TABLE table_name(bucket string, key string,
 version_id string) PARTITIONED BY (dt string)ROW FORMAT SERDE
 'org.apache.hadoop.hive.serde2.OpenCSVSerde'STORED AS INPUTFORMAT
 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat' OUTPUTFORMAT
 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat' LOCATION 'Copied S3 URI';
```

6. Um den Abfrage-Editor zu löschen, wählen Sie Löschen. Laden Sie den Inventory-Bericht mit dem folgenden Befehl in die Tabelle. Ersetzen Sie *table_name* durch denjenigen, den Sie im vorherigen Schritt ausgewählt haben. Wählen Sie dann Ausführen aus, um die Abfrage auszuführen.

```
MSCK REPAIR TABLE table_name;
```

7. Um den Abfrage-Editor zu löschen, wählen Sie Löschen. Führen Sie die folgende SELECT-Abfrage aus, um alle Einträge im ursprünglichen Inventory-Bericht abzurufen und alle leeren

Versions-IDs durch `null`-Zeichenfolgen zu ersetzen. Ersetzen Sie *`table_name`* durch das zuvor gewählte, und ersetzen Sie *`YYYY-MM-DD-HH-MM`* in der `WHERE`-Klausel durch das Datum des Inventory-Berichts, auf dem dieses Tool ausgeführt werden soll. Wählen Sie dann Ausführen aus, um die Abfrage auszuführen.

```
SELECT bucket as Bucket, key as Key, CASE WHEN version_id = '' THEN 'null' ELSE
 version_id END as VersionId FROM table_name WHERE dt = 'YYYY-MM-DD-HH-MM';
```

8. Kehren Sie zur Amazon-S3-Konsole (https://console.aws.amazon.com/s3/) zurück und navigieren Sie zu dem S3-Bucket, den Sie zuvor für den Standort des Abfrageergebnisses ausgewählt haben. Im Inneren sollte es eine Reihe von Ordnern geben, die mit dem Datum enden.

   Sie sollten beispielsweise etwas wie s3://***DOC-EXAMPLE-BUCKET***/***query-result-location***/Unsaved/2021/10/07/ sehen. Sie sollten `.csv`-Dateien sehen, die die Ergebnisse der ausgeführten `SELECT`-Abfrage enthalten.

   Wählen Sie die CSV-Datei mit dem letzten Änderungsdatum aus. Laden Sie diese Datei für den nächsten Schritt auf Ihren lokalen Computer herunter.

9. Die generierte CSV-Datei enthält eine Headerzeile. Um diese CSV-Datei als Eingabe für einen S3-Batchvorgangsauftrag zu verwenden, müssen Sie den Header entfernen, da der Batchvorgang keine Header auf CSV-Manifesten unterstützt.

   Um den Header zu entfernen, können Sie einen der folgenden Befehle für die Datei ausführen. Ersetzen Sie *`file.csv`* durch den Namen von Ihrer CSV-Datei.

   Führen Sie auf macOS- und Linux-Computern den `tail`-Befehl in einem Terminalfenster aus.

   ```
   tail -n +2 file.csv > tmp.csv && mv tmp.csv file.csv
   ```

   Führen Sie für Windows-Computer das folgende Skript in einem Windows-PowerShell-Fenster aus. Ersetzen Sie *`File-location`* durch den Dateipfad und *`file.csv`* durch den Namen Ihrer Datei.

   ```
   $ins = New-Object System.IO.StreamReader File-location\file.csv
   $outs = New-Object System.IO.StreamWriter File-location\temp.csv
   try {
       $skip = 0
       while ( !$ins.EndOfStream ) {
   ```

```
        $line = $ins.ReadLine();
        if ( $skip -ne 0 ) {
            $outs.WriteLine($line);
        } else {
            $skip = 1
        }
    }
} finally {
    $outs.Close();
    $ins.Close();
}
Move-Item File-location\temp.csv File-location\file.csv -Force
```

10. Nachdem Sie den Header aus der CSV-Datei entfernt haben, können Sie sie als Manifest in einem S3-Batchvorgangsauftrag verwenden. Laden Sie die CSV-Datei in einen S3-Bucket oder einen Speicherort Ihrer Wahl hoch und erstellen Sie dann einen Batchvorgang-Auftrag mit der CSV-Datei als Manifest.

    Weitere Informationen zum Erstellen eines S3-Batchvorgang-Auftrags finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags.

Arbeiten mit dem Feld „Objekt-ACL“

Eine Amazon-S3-Inventory-Bericht enthält eine Liste der Objekte im S3-Quell-Bucket sowie die Metadaten für jedes Objekt. Das Feld „Objekt-Zugriffsteuerungsliste (ACL)“ ist ein Metadatenfeld, das in Amazon S3 Inventory verfügbar ist. Insbesondere enthält das Feld „Objekt-ACL“ die Zugriffsteuerungsliste für jedes Objekt. Die ACL für ein Objekt definiert, welchen AWS-Konten oder Gruppen Zugriff auf dieses Objekt gewährt wird, und welche Art von Zugriff gewährt wird. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht und Amazon-S3-Inventory-Liste.

Das Feld „Objekt-ACL“ in Berichten von Amazon S3 Inventory ist im JSON-Format definiert. Die JSON-Daten enthalten die folgenden Felder:

- `version` – Die Version des Objekt-ACL-Feldformats in den Bestandsberichten. Das Datumsformat lautet `yyyy-mm-dd`.
- `status` – Mögliche Werte sind `AVAILABLE` oder `UNAVAILABLE`, um anzugeben, ob eine Objekt-ACL für ein Objekt verfügbar ist. Wenn der Status für die Objekt-ACL `UNAVAILABLE` lautet, lautet der Wert des Felds „Objekteigentümer“ im Bestandsbericht ebenfalls `UNAVAILABLE`.

- `grants` – Berechtigung-Berechtigungsempfänger-Paare, die den Berechtigungsstatus jedes Berechtigungsempfängers auflisten, der von der Objekt-ACL gewährt wird. Die verfügbaren Werte für einen Berechtigungsempfänger sind `CanonicalUser` und `Group`. Weitere Informationen zu den Berechtigungsempfängern finden Sie unter Berechtigungsempfänger in Zugriffssteuerungslisten.

  Für einen Berechtigungsempfänger des Typs `Group` umfasst ein Berechtigung-Berechtigungsempfänger-Paar die folgenden Attribute:

  - `uri` – Eine vordefinierte Amazon-S3-Gruppe.
  - `permission` – Die ACL-Berechtigungen, die für das Objekt erteilt wurden. Weitere Informationen finden Sie unter ACL-Berechtigungen für ein Objekt.
  - `type` – Den Typ `Group`, was bedeutet, dass es sich bei dem Berechtigungsempfänger um eine Gruppe handelt.

  Für einen Berechtigungsempfänger des Typs `CanonicalUser` umfasst ein Berechtigung-Berechtigungsempfänger-Paar die folgenden Attribute:

  - `canonicalId` – Eine verschleierte Form der AWS-Konto-ID. Weitere Informationen finden Sie unter Ermitteln der kanonischen Benutzer-ID für Ihr AWS-Konto.

    > **Note**
    >
    > Wenn ein Berechtigungsempfänger in einer ACL die E-Mail-Adresse eines AWS-Konto ist, verwendet S3 Inventory die `canonicalId` von diesem AWS-Konto und den Typ `CanonicalUser`, um diesen Berechtigungsempfänger zu spezifizieren. Weitere Informationen finden Sie unter Berechtigungsempfänger in Zugriffssteuerungslisten.

  - `permission` – Die ACL-Berechtigungen, die für das Objekt erteilt wurden. Weitere Informationen finden Sie unter ACL-Berechtigungen für ein Objekt.
  - `type` – Den Typ `CanonicalUser`, was bedeutet, dass es sich bei dem Berechtigungsempfänger um ein AWS-Konto handelt.

Das folgende Beispiel zeigt mögliche Werte für das Feld „Objekt-ACL“ im JSON-Format:

```
{
    "version": "2022-11-10",
    "status": "AVAILABLE",
    "grants": [{
```

```
        "uri": "http://acs.amazonaws.com/groups/global/AllUsers",
        "permission": "READ",
        "type": "Group"
    }, {
        "canonicalId": "example-canonical-id",
        "permission": "FULL_CONTROL",
        "type": "CanonicalUser"
    }]
}
```

Note

Das Feld „Objekt-ACL“ ist im JSON-Format definiert. In einem Bestandsbericht wird der Wert für das Feld „Objekt-ACL“ als Base64-codierte Zeichenfolge angezeigt. Angenommen, Sie haben das folgende Feld „Objekt-ACL“ im JSON-Format:

```
{
        "version": "2022-11-10",
        "status": "AVAILABLE",
        "grants": [{
            "canonicalId": "example-canonical-user-ID",
            "type": "CanonicalUser",
            "permission": "READ"
        }]
}
```

Das Feld „Objekt-ACL“ ist codiert und wird als die folgende Base64-codierte Zeichenfolge angezeigt:

```
eyJ2ZXJzaW9uIjoiMjAyMi0xMS0xMCIsInN0YXR1cyI6IkFWQUlMQUJMRSIsImdyYW50cyI6W3siY2Fub25pY2FsSW
```

Um für das Feld „Objekt-ACL“ den dekodierten Wert in JSON abzurufen, können Sie dieses Feld in Amazon Athena abfragen. Abfragebeispiele finden Sie unter Abfragen von Amazon S3 Inventory mit Amazon Athena.

Replizieren von Objekten

Die Replikation ermöglicht das automatische, asynchrone Kopieren von Objekten in Amazon-S3-Buckets. Buckets, die für die Objektreplikation konfiguriert sind, können demselben AWS-Konto

oder verschiedenen -Konten gehören. Sie können Objekte in einen einzelnen Ziel-Bucket oder in mehrere Ziel-Buckets replizieren. Die Ziel-Buckets können sich in einer anderen AWS-Regionen oder in derselben Region wie der Quell-Bucket befinden.

Um neue Objekte automatisch zu replizieren, während sie in den Bucket geschrieben werden, verwenden Sie die Live-Replikation, z. B. Cross-Region Replication (CRR, regionsübergreifende Replikation). Um vorhandene Objekte in einen anderen On-Demand-Bucket zu replizieren, verwenden Sie die S3-Batch-Replikation. Weitere Informationen zum Replizieren vorhandener Objekte finden Sie unter Wann die S3-Batch-Replikation verwendet wird.

Zum Aktivieren der CRR fügen Sie Ihrem Quell-Bucket eine Replikationskonfiguration hinzu. In der Minimalkonfiguration muss Folgendes angegeben sein:

- Ziel-Bucket(s), in dem/denen Amazon S3 die Objekte replizieren soll
- Eine AWS Identity and Access Management (IAM)-Rolle, die Amazon S3 annehmen kann, um Objekte in Ihrem Namen zu replizieren

Zusätzliche Konfigurationsoptionen sind verfügbar. Weitere Informationen finden Sie unter Zusätzliche Replikations-Konfigurationen.

Wenn Sie detaillierte Metriken für S3 Replication abrufen möchten, einschließlich der Metriken zur Anzahl der Replikationsregeln, können Sie Amazon S3 Storage Lens verwenden. S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Weitere Informationen finden Sie unter Verwenden von S3 Storage Lens zum Schutz Ihrer Daten. Eine vollständige Liste der Metriken finden Sie im Glossar der S3-Storage-Lens-Metriken.

Themen

- Replizieren bestehender Objekte mit S3-Batch-Replikation
- Zusätzliche Replikations-Konfigurationen
- Abrufen von Replikationsstatusinformationen
- Weitere Überlegungen

Gründe zur Verwendung der Replikation

Die Replikation unterstützt Sie bei Folgendem:

- Replikation von Objekten unter Beibehaltung von Metadaten – Sie können mithilfe der Replikation Kopien Ihrer Objekte erstellen, die alle Metadaten enthalten, z. B. die ursprünglichen Objekterstellungszeiten und Versions-IDs. Diese Funktion ist wichtig, wenn Sie sicherstellen müssen, dass Ihr Replikat mit dem Quellobjekt identisch ist.
- Replikation von Objekten in verschiedene Speicherklassen – Sie können mit der Replikation Objekte direkt in S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive oder eine andere Speicherklasse in den Ziel-Buckets platzieren. Sie können Ihre Daten auch in dieselbe Speicherklasse replizieren und Lebenszyklus-Konfigurationen für die Ziel-Buckets verwenden, um Ihre Objekte bei zunehmendem Alter in eine Speicherklasse für seltener genutzte Objekte zu verschieben.
- Beibehalten von Objektkopien unter unterschiedlicher Eigentümerschaft – Unabhängig davon, wer das Quellobjekt besitzt, können Sie Amazon S3 anweisen, die Replikateigentümerschaft in das zu ändern AWS-Konto , das den Ziel-Bucket besitzt. Diese Instance wird als Eigentümer-Überschreibungs-Option bezeichnet. Sie können diese Option nutzen, um den Zugriff auf Objektreplikate einzuschränken.
- Speichern von Objekten über mehrere AWS-Regionen – Um geografische Unterschiede bei der Aufbewahrung Ihrer Daten sicherzustellen, können Sie mehrere Ziel-Buckets für verschiedene festlegen AWS-Regionen. Diese Funktion kann Ihnen helfen, bestimmte Compliance-Anforderungen zu erfüllen.
- Replizieren von Objekten innerhalb von 15 Minuten – Um Ihre Daten innerhalb eines vorhersehbaren Zeitraums in derselben AWS-Region oder über verschiedene Regionen hinweg zu replizieren, können Sie S3 Replication Time Control (S3 RTC) verwenden. S3 RTC repliziert 99,99 Prozent der neuen in Amazon S3 gespeicherten Objekte innerhalb von 15 Minuten (gestützt auf ein Service Level Agreement). Weitere Informationen finden Sie unter the section called “Verwenden der S3-Replikationszeitkontrolle”.

- Synchronisieren Sie Buckets, replizieren Sie vorhandene Objekte und replizieren Sie zuvor fehlgeschlagene oder replizierte Objekte – Verwenden Sie die Batch-Replikation als On-Demand-Replikationsaktion, um Buckets zu synchronisieren und vorhandene Objekte zu replizieren. Weitere Informationen dazu, wann Sie die Batch-Replikation verwenden sollten, finden Sie unter Wann die S3-Batch-Replikation verwendet wird.
- Objekte replizieren und zu einem Bucket in einer anderen AWS-Region wechseln – Damit während der Datenreplikation alle Metadaten und Objekte zwischen Buckets synchron bleiben, verwenden Sie Regeln für die bidirektionale Replikation, bevor Sie die Failover-Kontrollen für Amazon S3 Multi-Region Access Points konfigurieren. Bidirektionale Replikationsregeln tragen dazu bei, sicherzustellen, dass wenn Daten in den S3-Bucket geschrieben werden, auf den der Datenverkehr bei einem Failover zurückgreift, diese Daten dann zurück in den Quell-Bucket repliziert werden.

> **Note**
>
> S3 RTC gilt nicht für die Batch-Replikation. Die Batch-Replikation ist ein On-Demand-Replikationsauftrag und kann mit S3-Batchvorgängen verfolgt werden. Weitere Informationen finden Sie unter Verfolgen von Auftragsstatus- und Abschluss.

Verwenden der regionsübergreifenden Replikation

Die regionsübergreifende Replikation (Cross-Region Replication, CRR) von S3 wird verwendet, um Objekte in Amazon-S3-Buckets in verschiedene zu kopieren AWS-Regionen. CRR kann Sie bei Folgendem unterstützen:

- Einhalten der Compliance-Anforderungen – Auch wenn Amazon S3 Ihre Daten standardmäßig in mehreren geografisch entfernten Availability Zones speichert, machen es die Compliance-Anforderungen möglicherweise erforderlich, Daten in noch größeren Entfernungen zu speichern. Um diese Anforderungen zu erfüllen, verwenden Sie die Replikation in mehreren Regionen, um Daten zwischen entfernten AWS-Regionen zu replizieren.
- Minimieren der Latenz – Wenn sich Ihre Kunden an zwei geografischen Standorten befinden, können Sie die Latenz beim Zugriff auf Objekte minimieren, indem Sie Objektkopien in verwalten AWS-Regionen , die geografisch näher an Ihren Benutzern liegen.
- Erhöhen der betrieblichen Effizienz – Wenn Sie über Datenverarbeitungs-Cluster in zwei verschiedenen verfügen AWS-Regionen , die denselben Satz von Objekten analysieren, können Sie Objektkopien in diesen Regionen aufbewahren.

Verwenden von Replikation innerhalb derselben Region

Mit der Replikation innerhalb derselben Region (Same-Region Replication, SRR) können Objekte in Amazon-S3-Buckets in derselben kopiert werden AWS-Region. SRR kann Sie bei Folgendem unterstützen:

- Aggregieren von Protokollen in einen einzelnen Bucket – Wenn Sie Protokolle in mehrere Buckets oder kontoübergreifend speichern, können Sie Protokolle ganz einfach in einen einzelnen Bucket innerhalb derselben Region replizieren. Dies ermöglicht eine einfachere Protokollverarbeitung an einem einzelnen Standort.
- Konfigurieren von Live-Replikation zwischen Produktions- und Testkonten – Wenn Sie oder Ihre Kunden Produktions- und Testkonten haben, die dieselben Daten nutzen, können Sie Objekte kontoübergreifend replizieren und dabei die Objektmetadaten erhalten.
- Einhaltung der Gesetze zur Datenhoheit – Möglicherweise müssen Sie mehrere Kopien Ihrer Daten in separaten AWS-Konten innerhalb einer bestimmten Region speichern. Mit der Replikation innerhalb derselben Region können Sie automatisch kritische Daten replizieren, wenn Ihre Daten aufgrund von Compliance-Regelungen im Land bleiben müssen.

Wann sollte die bidirektionale Replikation verwendet werden

- Erstellen von freigegebenen Datensätzen über mehrere AWS-Regionen hinweg – Mit der Synchronisierung von Replikaänderungen können Sie Metadatenänderungen wie Objektzugriffskontrolllisten (ACLs), Objekt-Tags oder Objektsperren für Replikationsobjekte einfach replizieren. Diese bidirektionale Replikation ist wichtig, wenn Sie alle Objekte und Objektmetadatenänderungen synchron halten möchten. Sie können die Synchronisierung von Replikaänderungen für eine neue oder bestehende Replikationsregel aktivieren, wenn Sie eine bidirektionale Replikation zwischen zwei oder mehr Buckets in derselben oder verschiedenen AWS-Regionen durchführen.
- Daten während des Failovers regionsübergreifend synchronisieren – Sie können Daten in Buckets zwischen synchronisieren, AWS-Regionen indem Sie bidirektionale Replikationsregeln mit S3 Cross-Region Replication (CRR) direkt von einem Multi-Region Access Point aus konfigurieren. Um eine fundierte Entscheidung darüber zu treffen, wann ein Failover eingeleitet werden soll, können Sie auch S3-Replikationsmetriken aktivieren, um die Replikation in Amazon CloudWatch, S3 Replication Time Control (S3 RTC) oder vom Multi-Region Access Point aus zu überwachen.

- Sorgen Sie für hohe Verfügbarkeit Ihrer Anwendung – Selbst im Fall einer regionalen Datenverkehrsunterbrechung können Sie bidirektionale Replikationsregeln verwenden, um alle Metadaten und Objekte während der Datenreplikation bucketübergreifend synchron zu halten.

Wann die S3-Batch-Replikation verwendet wird

Die Batch-Replikation repliziert vorhandene Objekte als On-Demand-Option in verschiedene Buckets. Im Gegensatz zur Live-Replikation können diese Aufträge nach Bedarf ausgeführt werden. Die Batch-Replikation unterstützt Sie bei Folgendem:

- Replizieren vorhandener Objekte – Sie können die Batch-Replikation verwenden, um Objekte zu replizieren, die dem Bucket hinzugefügt wurden, bevor die Replikation innerhalb derselben Region oder die Replikation in mehreren Regionen konfiguriert wurden.
- Replizieren Sie Objekte, bei denen die Replikation zuvor fehlgeschlagen hat – Sie können einen Batch-Replikationsauftrag filtern, um Objekte mit dem Replikationsstatus FEHLGESCHLAGEN zu replizieren.
- Replizieren von Objekten, die bereits repliziert wurden – Möglicherweise müssen Sie mehrere Kopien Ihrer Daten in separaten AWS-Konten oder AWS-Regionen speichern. Die Batch-Replikation kann vorhandene Objekte an neu hinzugefügte Ziele replizieren.
- Replizieren von Replikaten von Objekten, die aus einer Replikationsregel erstellt wurden – Replikationskonfigurationen erstellen Replikate von Objekten in Ziel-Buckets. Replikate von Objekten können nur mit Batch-Replikation repliziert werden.

Anforderungen für die Replikation

Für die Replikation ist Folgendes erforderlich:

- Für das Konto des Quell-Bucket-Eigentümers müssen Quelle und Ziel AWS-Regionen aktiviert sein. Für das Konto des Ziel-Bucket-Eigentümers muss die Zielregion aktiviert sein.

  Weitere Informationen zum Aktivieren oder Deaktivieren eines AWS-Region finden Sie unter Verwalten AWS-Regionen von im Allgemeine AWS-Referenz.
- Für Quell- und Ziel-Buckets muss die Versioning aktiviert sein. Weitere Informationen über das Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

- Amazon S3 muss berechtigt sein, Objekte aus dem Quell-Bucket in Ihrem Namen in den Ziel-Bucket/die Ziel-Buckets zu replizieren. Weitere Informationen zu diesen Berechtigungen finden Sie unter Einrichten von Berechtigungen.
- Wenn der Eigentümer des Quell-Buckets das Objekt im Bucket nicht besitzt, muss der Objekteigentümer dem Bucket-Eigentümer in der Access Control List (ACL) des Objekts die Berechtigungen `READ` und `READ_ACP` gewähren. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.
- Wenn für den Quell-Bucket die S3-Objektsperre aktiviert ist, muss sie für die Ziel-Buckets ebenfalls aktiviert sein.

  Um die Replikation für einen Bucket zu aktivieren, für den die Objektsperre aktiviert ist, müssen Sie die AWS Command Line Interface, die REST-API oder AWS SDKs verwenden. Weitere allgemeine Informationen finden Sie unter Verwenden der S3-Objektsperre.

  > **Note**
  >
  > Sie müssen zwei neue Berechtigungen für den S3-Quell-Bucket in der AWS Identity and Access Management (IAM)-Rolle erteilen, mit der Sie die Replikation eingerichtet haben. Die zwei neuen Berechtigungen lauten `s3:GetObjectRetention` und `s3:GetObjectLegalHold`. Wenn die Rolle über eine `s3:Get*`-Berechtigung verfügt, ist die Anforderung erfüllt. Weitere Informationen finden Sie unter Einrichten von Berechtigungen.

Weitere Informationen finden Sie unter Einrichten der Replikation.

Wenn Sie die Replikationskonfiguration in einem kontoübergreifenden Szenario festlegen, in dem sich die Quell- und Ziel-Buckets im Besitz von verschiedenen AWS-Konten befinden, gilt die folgenden zusätzliche Anforderung:

- Der Eigentümer der Ziel-Buckets muss dem Eigentümer des Quell-Buckets mit einer Bucket-Richtlinie die Berechtigungen zum Replizieren von Objekten erteilen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten.
- Die Ziel-Buckets dürfen nicht als Buckets des Typs "Zahlung durch den Anforderer" konfiguriert sein. Weitere Informationen finden Sie unter Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung.

Was repliziert Amazon S3?

Amazon S3 repliziert nur bestimmte Elemente in Buckets, die für die Replikation konfiguriert sind.

Themen

Was wird mit Replikationskonfigurationen repliziert?

Standardmäßig repliziert Amazon S3 Folgendes:

- Objekte, die nach dem Hinzufügen einer Replikations-Konfiguration erstellt wurden.
- Unverschlüsselte Objekte.
- Objekte, die mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verschlüsselt wurden, Objekte, die im Ruhezustand mit einem von Amazon S3 verwalteten Schlüssel (SSE-S3) oder einem in gespeicherten KMS-Schlüssel AWS Key Management Service (SSE-KMS) verschlüsselt wurden. Weitere Informationen finden Sie unter the section called “Replizieren von verschlüsselten Objekten (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS)”.
- Objektmetadaten von den Quellobjekten zu den Replikaten Informationen zum Replizieren von Metadaten aus den Replikaten zu den Quellobjekten finden Sie unter Replizieren von Metadatenänderungen mit der Synchronisierung von Amazon-S3-Replikatänderungen.
- Nur die Objekte im Quell-Bucket, für die der Bucket-Eigentümer über Berechtigungen zum Lesen von Objekten und von ACLs verfügt.

  Weitere Informationen zum Eigentum an Ressourcen finden Sie unter Amazon-S3-Bucket- und Objekt-Eigentümerschaft.
- Aktualisierungen der Objekt-ACL, es sei denn, Sie weisen Amazon S3 an, die Replikat-Eigentümerschaft zu ändern, wenn sich Quell- und Ziel-Buckets nicht im Besitz derselben Konten befinden

  Weitere Informationen finden Sie unter Ändern des Replikat-Eigentümers.

  Es kann ein wenig dauern, bis Amazon S3 die beiden ACLs synchronisiert hat. Diese Änderung der Eigentümerschaft gilt nur für Objekte, die erstellt wurden, nachdem Sie dem Bucket eine Replikationskonfiguration hinzugefügt haben.

- Objekt-Markierungen, sofern vorhanden.
- Informationen zur Beibehaltung der S3-Objektsperre, sofern vorhanden.

  Wenn Amazon S3 Objekte mit angewandten Aufbewahrungsinformationen repliziert, wendet es dieselben Aufbewahrungssteuerungen auf Ihre Replikate an, wodurch der für Ihre Ziel-Buckets konfigurierte standardmäßige Aufbewahrungszeitraum überschrieben wird. Wenn auf die Objekte in Ihrem Quell-Bucket keine Aufbewahrungssteuerungen angewandt sind und Sie sie in Ziel-Buckets mit festgelegtem Standard-Aufbewahrungszeitraum replizieren, wird der Standard-Aufbewahrungszeitraum der Ziel-Buckets auf Ihre Objektreplikate angewandt. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

Auswirkungen von LöschVorgänge auf die Replikation

Wenn Sie ein Objekt aus dem Quell-Bucket löschen, werden standardmäßig die folgenden Aktionen ausgeführt:

- Wenn Sie eine DELETE-Anforderung ohne Angabe einer Objektversions-ID stellen, fügt Amazon S3 eine Löschmarkierung hinzu. Amazon S3 befasst sich wie folgt mit der Löschmarkierung:
  - Wenn Sie die aktuelle Version der Replikations-Konfiguration verwenden, d. h. das Element `Filter` in einer Replikations-Konfigurations-Regel angeben, repliziert Amazon S3 die Löschmarkierung nicht standardmäßig. Sie können Regeln jedoch die Replikation von Löschmarkierungen non-tag-based hinzufügen. Weitere Informationen finden Sie unter Replizieren von Löschmarkierungen auf Buckets.
  - Wenn Sie das Element `Filter` nicht angeben, geht Amazon S3 davon aus, dass die Replikations-Konfiguration Version V1 ist, und repliziert Löschmarkierungen, die aus Benutzeraktionen resultierten. Wenn Amazon S3 jedoch ein Objekt aufgrund einer Lebenszyklus-Aktion löscht, wird die Löschmarkierung nicht auf die Ziel-Buckets repliziert.
- Wenn Sie angeben, dass eine Objektversions-ID in einer `DELETE`-Anforderung gelöscht werden soll, löscht Amazon S3 diese Objektversion im Quell-Bucket. Die Löschung wird jedoch nicht in den Ziel-Buckets repliziert. Anders ausgedrückt: Dieselbe Objektversion wird aus den Ziel-Buckets nicht gelöscht. Dies schützt Daten vor missbräuchlichen Löschungen.

Was wird mit Replikationskonfigurationen nicht repliziert?

Standardmäßig repliziert Amazon S3 Folgendes nicht:

- Objekte im Quell-Bucket, bei denen es sich um Replikate handelt, die von einer anderen Replikationsregel erstellt wurden. Zum Beispiel: Angenommen Sie konfigurieren eine Replikation, bei der Bucket A die Quelle und Bucket B das Ziel ist. Nehmen wir jetzt an, Sie fügen eine weitere Replikations-Konfiguration hinzu, bei der Bucket B die Quelle und Bucket C das Ziel ist. In diesem Fall werden Objekte in Bucket B, die Replikate von Objekten in Bucket A sind, nicht in Bucket C repliziert.

  Verwenden Sie die Batch-Replikation, um Objekte zu replizieren, die Replikate sind. Weitere Informationen zum Konfigurieren der Batch-Replikation finden Sie unter Replizieren vorhandener Objekte.
- Objekte im Quell-Bucket, die bereits auf ein anderes Ziel repliziert wurden. Wenn Sie beispielsweise den Ziel-Bucket in einer vorhandenen Replikationskonfiguration ändern, repliziert Amazon S3 diese Objekte nicht erneut.

  Verwenden Sie die Batch-Replikation, um zuvor replizierte Objekte zu replizieren. Weitere Informationen zum Konfigurieren der Batch-Replikation finden Sie unter Replizieren vorhandener Objekte.
- Die Batch-Replikation unterstützt nicht das erneute Replizieren von Objekten, die mit der Versions-ID des Objekts aus dem Ziel-Bucket gelöscht wurden. Wenn Sie diese Objekte erneut replizieren möchten, können Sie die Quellobjekte mit einem Batch–Kopierauftrag kopieren. Beim Kopieren dieser Objekte werden neue Versionen des Objekts im Quell-Bucket erstellt und die Replikation zum Ziel wird automatisch initiiert. Weitere Informationen zur Verwendung von Batch Copy finden Sie unter Beispiele, die Batch-Vorgänge zum Kopieren von Objekten verwenden.
- Bei der Replikation aus einem anderen werden AWS-Konto Löschmarkierungen, die dem Quell-Bucket hinzugefügt wurden, standardmäßig nicht repliziert.

  Informationen zum Replizieren von Löschmarkierungen finden Sie unter Replizieren von Löschmarkierungen auf Buckets.
- Objekte, die in den Speicherklassen oder Stufen S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive, S3 Intelligent-Tiering Archive Access oder S3 Intelligent-Tiering Deep Archive Access gespeichert sind. Sie können diese Objekte erst replizieren, wenn Sie sie wiederhergestellt und in eine andere Speicherklasse kopiert haben.

  Weitere Informationen zu S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive finden Sie unter Speicherklassen für die Archivierung von Objekten.

  Weitere Informationen zu S3 Intelligent-Tiering finden Sie unter Amazon S3 Intelligent Tiering.

- Objekte im Quell-Bucket, für die der Bucket-Eigentümer nicht über ausreichende Berechtigungen zur Replikation verfügt.

  Weitere Informationen darüber, wie ein Objekt-Eigentümer einem Bucket-Eigentümer Berechtigungen erteilen kann, finden Sie unter Erteilung von kontoübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt.

- Aktualisierungen von Unterressourcen auf Bucket-Ebene.

  Wenn Sie beispielsweise die Lebenszyklus-Konfiguration ändern oder eine Benachrichtigungskonfiguration zu Ihrem Quell-Bucket hinzufügen, werden diese Änderungen nicht auf den Ziel-Bucket angewendet. Durch diese Funktion ist es möglich, für den Quell- und den Ziel-Bucket verschiedene Konfigurationen zu nutzen.

- Aktionen, die von der Lebenszyklus-Konfiguration durchgeführt werden.

  Wenn eine Lebenszyklus-Konfiguration beispielsweise nur auf Ihrem Quell-Bucket aktiviert ist, erstellt Amazon S3 Löschmarkierungen für abgelaufene Objekte, repliziert diese Markierungen jedoch nicht. Wenn Sie dieselbe Lebenszyklus-Konfiguration sowohl auf den Quell- als auch auf den Ziel-Bucket anwenden möchten, aktivieren Sie für beide Buckets dieselbe Lebenszyklus-Konfiguration. Weitere Informationen zur Lebenszyklus-Konfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Standard-Bucket-Verschlüsselung und Replikation

Wenn Sie die Standard-Verschlüsselung für einen Replikations-Ziel-Bucket aktivieren, gilt das folgende Verschlüsselungsverhalten:

- Wenn Objekte im Quell-Bucket nicht verschlüsselt sind, werden die Replikatobjekte im Ziel-Bucket mithilfe der Einstellungen der Standard-Verschlüsselung des Ziel-Buckets verschlüsselt. Daher unterscheiden sich die ETags (Entity-Tags) der Quellobjekte von den ETags der Replikatobjekte. Wenn Sie Anwendungen haben, die ETags verwenden, müssen Sie diese Anwendungen aktualisieren, um diesen Unterschied auszugleichen.
- Wenn Objekte im Quell-Bucket mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3), serverseitiger Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder serverseitiger Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) verschlüsselt werden, verwenden die Replikatobjekte im Ziel-

Bucket denselben Verschlüsselungstyp wie die Quellobjekte. Die Einstellungen der Standard-Verschlüsselung des Ziel-Buckets werden nicht verwendet.

Einrichten der Replikation

> **Note**
>
> Objekte, die vor dem Einrichten der Replikation vorhanden waren, werden nicht automatisch repliziert. Anders ausgedrückt: Amazon S3 repliziert Objekte nicht rückwirkend. Verwenden Sie die S3-Batch-Replikation, um Objekte zu replizieren, die vor Ihrer Replikationskonfiguration erstellt wurden. Weitere Informationen zum Konfigurieren der Batch-Replikation finden Sie unter Replizieren vorhandener Objekte.

Um die Replikation in derselben Region (SRR) oder Replikation in mehreren Regionen (CRR) zu aktivieren, fügen Sie Ihrem Quell-Bucket eine Replikationskonfiguration hinzu. Die Konfiguration weist Amazon S3 an, Objekte wie angegeben zu replizieren. In der Replikations-Konfiguration müssen Sie Folgendes angeben:

- Die Ziel-Buckets – Der Bucket oder die Buckets, in den/die Amazon S3 die Objekte replizieren soll.
- Die Objekte, die Sie replizieren möchten – Sie können alle Objekte im Quell-Bucket replizieren oder nur eine Teilmenge davon. Teilmengen identifizieren Sie, indem Sie ein Schlüsselnamenpräfix, mindestens ein Objekt-Tag oder beides in der Konfiguration angeben.

  Wenn Sie beispielsweise eine Replikationsregel konfigurieren, um nur Objekte mit dem Schlüsselnamenpräfix `Tax/` zu replizieren, repliziert Amazon S3 Objekte mit Schlüsseln wie `Tax/doc1` oder `Tax/doc2`. Es repliziert aber keine Objekte mit dem Schlüssel `Legal/doc3`. Wenn Sie sowohl ein Präfix als auch mindestens ein Tag angeben, repliziert Amazon S3 nur Objekte, die dieses Schlüsselpräfix und diese Tags aufweisen.

Zusätzlich zu diesen Mindestanforderungen können Sie die folgenden Optionen auswählen:

- Replikat-Speicherklasse – Standardmäßig speichert Amazon S3 Objektreplikate in derselben Speicherklasse wie das Quellobjekt. Sie können eine andere Speicherklasse für die Replikate festlegen.
- Replikat-Eigentümerschaft – Amazon S3 geht davon aus, dass ein Objektreplikat weiterhin das Eigentum des Eigentümers des Quellobjekts bleibt. Wenn es Objekte repliziert, wird auch die

entsprechende Objekt-Zugriffssteuerungsliste (ACL) oder die S3-Object-Ownership-Einstellung repliziert. Wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener AWS-Konten befinden, können Sie die Replikation so konfigurieren, dass der Besitzer eines Replikats auf das AWS-Konto geändert wird, dem der Ziel-Bucket gehört.

Sie können die Replikation mithilfe der REST API, AWS SDKs , AWS Command Line Interface (AWS CLI) oder der Amazon S3-Konsole konfigurieren.

Amazon S3 stellt auch API-Vorgänge zur Unterstützung der Einrichtung von Replikationsregeln bereit. Weitere Informationen finden Sie in den folgenden Themen in der Amazon Simple Storage Service – API-Referenz.

- PUT Bucket replication
- GET Bucket replication
- DELETE Bucket replication

Themen

- Replikations-Konfiguration
- Einrichten von Berechtigungen
- Anleitungen: Beispiele zum Konfigurieren der Replikation

Replikations-Konfiguration

Amazon S3 speichert die Replikations-Konfiguration als XML. In der XML-Datei für die Replikationskonfiguration geben Sie eine AWS Identity and Access Management (IAM)-Rolle und eine oder mehrere Regeln an.

```
<ReplicationConfiguration>
    <Role>IAM-role-ARN</Role>
    <Rule>
        ...
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
</ReplicationConfiguration>
```

Amazon S3 kann ohne Berechtigung durch Sie keine Objekte replizieren. Sie gewähren Berechtigungen mit der IAM-Rolle, die Sie in der Replikations-Konfiguration angeben. Amazon S3 übernimmt die IAM-Rolle, um Objekte in Ihrem Namen zu replizieren. Sie müssen der IAM-Rolle zunächst die erforderlichen Berechtigungen erteilen. Weitere Informationen zum Verwalten von Berechtigungen finden Sie unter Einrichten von Berechtigungen.

In den folgenden Szenarien fügen Sie eine Regel zur Replikationskonfiguration hinzu:

- Sie möchten alle Objekte replizieren.
- Sie möchten eine Teilmenge der Objekte replizieren. Sie identifizieren die Teilmenge der Objekte, indem Sie einen Filter zur Regel hinzufügen. In dem Filter geben Sie ein Objektschlüsselpräfix, Markierungen oder eine Kombination aus beidem an, um die Objektteilmenge zu identifizieren, für die die Regel gilt. Die Filter zielen auf Objekte ab, die genau den von Ihnen angegebenen Werten entsprechen.

Sie fügen mehrere Regeln zu einer Replikationskonfiguration hinzu, wenn Sie eine andere Teilmenge von Objekten replizieren möchten. In jeder Regel geben Sie einen Filter an, der eine andere Teilmenge von Objekten auswählt. Beispiel: Sie möchten Objekte mit dem Schlüsselpräfix `tax/` oder `document/` replizieren. Dazu fügen Sie zwei Regeln hinzu, eine, die den `tax/`-Schlüsselpräfix-Filter angibt und eine andere, die das `document/`-Schlüsselpräfix angibt. Weitere Hinweise zum Präfix für Objektschlüssel finden Sie unter Organisieren von Objekten mit Präfixen.

In den folgenden Abschnitten finden Sie zusätzliche Informationen.

Themen

Basisregelkonfiguration

Jede Regel muss den Status und die Priorität der Regel enthalten. Die Regel muss auch angeben, ob Löschmarkierungen repliziert werden sollen.

- `Status` gibt an, ob die Regel mithilfe der Werte `Enabled` oder `Disabled` aktiviert oder deaktiviert ist. Wenn eine Regel deaktiviert ist, führt Amazon S3 die in der Regel angegebenen Aktionen nicht durch.
- `Priority` gibt an, welche Regel Vorrang hat, wenn zwei oder mehr Replikationsregeln in Konflikt stehen. Amazon S3 versucht, Objekte gemäß allen Replikationsregeln zu replizieren. Wenn es jedoch zwei oder mehr Regeln mit demselben Ziel-Bucket gibt, werden Objekte gemäß der Regel mit der höchsten Priorität repliziert. Je höher die Zahl, desto höher die Priorität.
- `DeleteMarkerReplication` gibt an, ob Löschmarkierungen unter Verwendung der Werte `Enabled` oder `Disabled` repliziert werden sollen.

In der Zielkonfiguration müssen Sie den Namen des/der Buckets angeben, in den/die Amazon S3 Objekte replizieren soll.

Das folgende Beispiel zeigt die Mindestanforderungen für eine V2-Regel. Aus Gründen der Abwärtskompatibilität unterstützt Amazon S3 weiterhin das Format XML V1. Weitere Informationen finden Sie unter Abwärtskompatibilität.

```
...
    <Rule>
        <ID>Rule-1</ID>
        <Status>Enabled-or-Disabled</Status>
        <Filter>
            <Prefix></Prefix>
        </Filter>
        <Priority>integer</Priority>
        <DeleteMarkerReplication>
           <Status>Enabled-or-Disabled</Status>
        </DeleteMarkerReplication>
        <Destination>
           <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket>
        </Destination>
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
...
```

Sie können auch andere Konfigurationsoptionen festlegen. Beispiel: Sie möchten für Objektreplikate eine andere Speicherklasse verwenden als für das Quellobjekt.

Optional: Festlegen eines Filters

Um eine Teilmenge von Objekten auszuwählen, für die die Regel gilt, fügen Sie einen optionalen Filter hinzu. Sie können nach Objekt-Schlüsselpräfix, Objekt-Tags oder einer Kombination aus beidem filtern. Wenn Sie sowohl nach Schlüsselpräfix als auch nach Objekt-Tags filtern, kombiniert Amazon S3 die Filter mit einem logischen AND-Operator. Anders ausgedrückt: Die Regel wird auf eine Teilmenge von Objekten mit einem bestimmten Schlüsselpräfix und bestimmten Markierungen angewendet.

Filter basierend auf Objektschlüsselpräfix

Um eine Regel mit einem Filter nach Objektschlüsselpräfix festzulegen, verwenden Sie den folgenden Code. Sie können nur ein Präfix angeben.

```
<Rule>
    ...
    <Filter>
        <Prefix>key-prefix</Prefix>
    </Filter>
    ...
</Rule>
...
```

Filter basierend auf Objekt-Markierungen

Um eine Regel mit einem Filter nach Objekt-Markierungen festzulegen, verwenden Sie den folgenden Code. Sie können einen oder mehrere Objekt-Markierungen festlegen.

```
<Rule>
    ...
    <Filter>
        <And>
            <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
            </Tag>
            <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
            </Tag>
             ...
```

```
        </And>
    </Filter>
    ...
</Rule>
...
```

Filtern mit einem Schlüsselpräfix und Objekt-Markierungen

Um eine Regel mit einem Filter nach einer Kombination aus Schlüsselpräfix und Objekt-Markierungen festzulegen, verwenden Sie den folgenden Code. Sie wickeln diese Filter in ein übergeordnetes And-Element ein. Amazon S3 führt eine logische AND-Operation aus, um diese Filter zu kombinieren. Anders ausgedrückt: Die Regel wird auf eine Teilmenge von Objekten sowohl mit einem bestimmten Schlüsselpräfix als auch mit bestimmten Tags angewendet.

```
<Rule>
    ...
    <Filter>
        <And>
            <Prefix>key-prefix</Prefix>
            <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
            </Tag>
            <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
            </Tag>
             ...
    </Filter>
    ...
</Rule>
...
```

> **Note**
>
> Wenn Sie eine Regel mit einem leeren Filter-Tag angeben, gilt Ihre Regel für alle Objekte in Ihrem Bucket.

Zusätzliche Zielkonfigurationen

In der Zielkonfiguration geben Sie den/die Bucket(s) an, in den/die Amazon S3 Objekte replizieren soll. Sie können Konfigurationen einrichten, um Objekte aus einem Quell-Bucket in einen oder mehrere Ziel-Buckets zu replizieren.

```
...
<Destination>
    <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket>
</Destination>
...
```

Sie können die folgenden Optionen im Element `<Destination>` hinzufügen.

Themen

- Festlegen der Speicherklasse
- Hinzufügen mehrerer Ziel-Buckets
- Angeben verschiedener Parameter für jede Replikationsregel mit mehreren Ziel-Buckets
- Ändern der Replikat-Eigentümerschaft
- Aktivieren der S3 Replication Time Control
- Replizieren von mit serverseitiger Verschlüsselung erstellten Objekten mithilfe von AWS KMS

Festlegen der Speicherklasse

Sie können die Speicherklasse für die Objektreplikate festlegen. Standardmäßig verwendet Amazon S3 wie im folgenden Beispiel die Speicherklasse des Quellobjekts zum Erstellen von Objektreplikaten.

```
...
<Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket>
       <StorageClass>storage-class</StorageClass>
</Destination>
...
```

Hinzufügen mehrerer Ziel-Buckets

Sie können wie folgt mehrere Ziel-Buckets in einer einzigen Replikations-Konfiguration hinzufügen.

```
...
<Rule>
    <ID>Rule-1</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Enabled-or-Disabled</Status>
    </DeleteMarkerReplication>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
</Rule>
<Rule>
    <ID>Rule-2</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Enabled-or-Disabled</Status>
    </DeleteMarkerReplication>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET2</Bucket>
    </Destination>
</Rule>
...
```

Angeben verschiedener Parameter für jede Replikationsregel mit mehreren Ziel-Buckets

Wenn Sie mehrere Ziel-Buckets in einer einzigen Replikations-Konfiguration hinzufügen, können Sie wie folgt verschiedene Parameter für jede Replizierungsregel angeben.

```
...
<Rule>
    <ID>Rule-1</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Disabled</Status>
    </DeleteMarkerReplication>
      <Metrics>
    <Status>Enabled</Status>
    <EventThreshold>
      <Minutes>15</Minutes>
    </EventThreshold>
```

```
  </Metrics>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
</Rule>
<Rule>
    <ID>Rule-2</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Enabled</Status>
    </DeleteMarkerReplication>
      <Metrics>
    <Status>Enabled</Status>
    <EventThreshold>
      <Minutes>15</Minutes>
    </EventThreshold>
  </Metrics>
  <ReplicationTime>
    <Status>Enabled</Status>
    <Time>
      <Minutes>15</Minutes>
    </Time>
  </ReplicationTime>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET2</Bucket>
    </Destination>
</Rule>
...
```

Ändern der Replikat-Eigentümerschaft

Wenn sich der Quell- und der Ziel-Bucket nicht im Besitz desselben -Kontos befinden, können Sie die Eigentümerschaft des Replikats auf das ändern AWS-Konto , das den Ziel-Bucket besitzt. Fügen Sie dazu das `AccessControlTranslation`-Element hinzu. Dieses Element übernimmt den Wert `Destination`.

```
...
<Destination>
   <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket>
   <Account>destination-bucket-owner-account-id</Account>
   <AccessControlTranslation>
       <Owner>Destination</Owner>
```

```
   </AccessControlTranslation>
</Destination>
...
```

Wenn Sie das -`AccessControlTranslation`Element nicht zur Replikationskonfiguration hinzufügen, gehören die Replikate demselben AWS-Konto , dem das Quellobjekt gehört. Weitere Informationen finden Sie unter Ändern des Replikat-Eigentümers.

Aktivieren der S3 Replication Time Control

Sie können die S3-Replikationszeitkontrolle (S3 RTC) in Ihrer Replikations-Konfiguration aktivieren. S3 RTC repliziert die meisten Objekte in Sekunden und 99,99 Prozent der Objekte innerhalb von 15 Minuten (gestützt auf ein Service Level Agreement).

> **Note**
>
> Nur ein Wert von `<Minutes>15</Minutes>` wird für `EventThreshold` und `Time` akzeptiert.

```
...
<Destination>
  <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket>
  <Metrics>
    <Status>Enabled</Status>
    <EventThreshold>
      <Minutes>15</Minutes>
    </EventThreshold>
  </Metrics>
  <ReplicationTime>
    <Status>Enabled</Status>
    <Time>
      <Minutes>15</Minutes>
    </Time>
  </ReplicationTime>
</Destination>
...
```

Weitere Informationen finden Sie unter Erfüllen der Compliance-Anforderungen mit S3-Replikationszeitkontrolle (S3 RTC). API-Beispiele finden Sie unter PutBucketReplication in der API-Referenz zu Amazon Simple Storage Service.

Replizieren von mit serverseitiger Verschlüsselung erstellten Objekten mithilfe von AWS KMS

Ihr Quell-Bucket enthält möglicherweise Objekte, die mit serverseitiger Verschlüsselung mithilfe von AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) erstellt wurden. Standardmäßig repliziert Amazon S3 diese Objekte nicht. Optional können Sie Amazon S3 zum Replizieren dieser Objekte anweisen. Um dies zu tun, entscheiden Sie sich zunächst explizit für diese Funktion, indem Sie das `SourceSelectionCriteria`-Element hinzufügen. Geben Sie dann die AWS KMS key (für die AWS-Region des Ziel-Buckets) an, die zum Verschlüsseln von Objektreplikaten verwendet werden soll. Das folgende Beispiel zeigt, wie diese Elemente angegeben werden.

```
...
<SourceSelectionCriteria>
  <SseKmsEncryptedObjects>
    <Status>Enabled</Status>
  </SseKmsEncryptedObjects>
</SourceSelectionCriteria>
<Destination>
  <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket>
  <EncryptionConfiguration>
    <ReplicaKmsKeyID>AWS KMS key ID to use for encrypting object replicas</
ReplicaKmsKeyID>
  </EncryptionConfiguration>
</Destination>
...
```

Weitere Informationen finden Sie unter Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten.

Beispiele für Replikations-Konfigurationen

Fügen Sie zunächst die folgenden Beispiel-Replikations-Konfigurationen zu Ihrem Bucket hinzu.

> **Important**
>
> Um eine Replikationskonfiguration zu einem Bucket hinzuzufügen, benötigen Sie die `iam:PassRole`-Berechtigung. Diese Berechtigung erlaubt es Ihnen, die IAM-Rolle weiterzugeben, die Amazon S3 die Replikationsberechtigungen erteilt. Sie legen die IAM-Rolle fest, indem Sie den Amazon-Ressourcennamen (ARN) angeben, der im `Role`-Element in der XML-Datei der Replikationskonfiguration verwendet wird. Weitere Informationen finden

Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service-Service übergeben kann im IAM-Benutzerhandbuch.

Example 1: Replikations-Konfiguration mit einer Regel

Die folgende grundlegende Replikations-Konfiguration legt eine Regel fest. Die Regel legt eine IAM-Rolle, die Amazon S3 annehmen kann, sowie einen einzigen Ziel-Bucket für Objektreplikate fest. Der `Status`-Wert von `Enabled` gibt an, dass die Regel aktiviert ist.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>

    <Destination><Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

Um eine Teilmenge von Objekten für die Replikation auszuwählen, können Sie einen Filter hinzufügen. In der folgenden Konfiguration gibt der Filter ein Objektschlüsselpräfix an. Diese Regel gilt für Objekte mit dem Schlüsselnamenpräfix *Tax/*.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>

    <Filter>
       <Prefix>Tax/</Prefix>
    </Filter>

    <Destination><Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket></Destination>

  </Rule>
```

```
</ReplicationConfiguration>
```

Wenn Sie das `Filter`-Element angeben, müssen Sie auch die Elemente `Priority` und `DeleteMarkerReplication` einschließen. In diesem Beispiel ist die `Priority` (Priorität) nicht relevant, da nur eine Regel vorhanden ist.

In der folgenden Konfiguration gibt der Filter ein Präfix und zwei Markierungen an. Die Regel gilt für eine Untermenge der Objekte, die das angegebene Schlüsselpräfix und die angegebenen Markierungen aufweisen. Insbesondere gilt es für ein Objekt, das das *Tax/*-Präfix in seinen Schlüsselnamen und den beiden angegebenen Objekt-Tags hat. Die `Priority` (Priorität) trifft nicht zu, weil es nur eine Regel gibt.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>

    <Filter>
        <And>
          <Prefix>Tax/</Prefix>
          <Tag>
             <Tag>
                <Key>tagA</Key>
                <Value>valueA</Value>
             </Tag>
          </Tag>
          <Tag>
             <Tag>
                <Key>tagB</Key>
                <Value>valueB</Value>
             </Tag>
          </Tag>
       </And>

    </Filter>

    <Destination><Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket></Destination>
```

```
  </Rule>
</ReplicationConfiguration>
```

Sie können eine Speicherklasse für die Objektreplikate wie folgt festlegen.

```
<?xml version="1.0" encoding="UTF-8"?>

<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket>
       <StorageClass>storage-class</StorageClass>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Sie können jede Speicherklasse festlegen, die Amazon S3 unterstützt.

Example 2: Replikations-Konfiguration mit zwei Regeln

Example

In der folgenden Replikations-Konfiguration:

- Jede Regel filtert nach einem anderen Schlüsselpräfix, sodass jede Regel auf eine bestimmte Teilmenge von Objekten angewendet wird. In diesem Beispiel repliziert Amazon S3 Objekte mit den Schlüsselnamen *Tax/doc1.pdf* und *Project/project1.txt*, repliziert jedoch keine Objekte mit dem Schlüsselnamen *PersonalDoc/documentA*.
- Die Regelpriorität ist nicht relevant, da die Regeln für zwei unterschiedliche Mengen an Objekten gelten. Das nächste Beispiel zeigt, was geschieht, wenn die Regelpriorität angewendet wird.
- Die zweite Regel gibt eine Speicherklasse S3 Standard-IA für Objektreplikate an. Amazon S3 verwendet die angegebene Speicherklasse für diese Objektreplikate.

```
<?xml version="1.0" encoding="UTF-8"?>

<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
```

```
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>Tax</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
     ...
  </Rule>
 <Rule>
    <Status>Enabled</Status>
    <Priority>2</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>Project</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
     <StorageClass>STANDARD_IA</StorageClass>
    </Destination>
     ...
  </Rule>


</ReplicationConfiguration>
```

Example 3: Replikations-Konfiguration mit zwei Regeln mit einander überlappenden Präfixen

In dieser Konfiguration geben die zwei Regeln Filter mit einander überlappenden Schlüsselpräfixen an – *star/* und *starship/*. Beide Regeln gelten für Objekte mit dem Schlüsselnamen *starship-x*. In diesem Fall nutzt Amazon S3 die Regelpriorität, um zu bestimmen, welche Regel angewendet werden soll. Je höher die Zahl, desto höher die Priorität.

```
<ReplicationConfiguration>
```

```
  <Role>arn:aws:iam::account-id:role/role-name</Role>

  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>star</Prefix>
    </Filter>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
  </Rule>
  <Rule>
    <Status>Enabled</Status>
    <Priority>2</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>starship</Prefix>
    </Filter>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Example 4. Beispielhafte Walkthroughs

Beispielanleitungen finden Sie unter Anleitungen: Beispiele zum Konfigurieren der Replikation.

Weitere Informationen zur XML-Struktur der Replikationskonfiguration finden Sie unter PutBucketReplication in der API-Referenz zu Amazon Simple Storage Service.

Abwärtskompatibilität

Die aktuelle Version der Replikations-Konfigurations-XML ist V2. XML V2-Replikations-Konfigurationen sind solche, die das `Filter`-Element für Regeln und Regeln enthalten, die S3-Replikations-Zeitkontrolle (S3 RTC) angeben.

Um die Version Ihrer Replikationskonfiguration anzuzeigen, können Sie den `GetBucketReplication`-API-Vorgang verwenden. Weitere Informationen finden Sie unter GetBucketReplication in der API-Referenz zu Amazon Simple Storage Service.

Aus Gründen der Abwärtskompatibilität unterstützt Amazon S3 weiterhin die XML-V1-Replikations-Konfiguration. Wenn Sie die Replikationskonfiguration XML V1 verwendet haben, berücksichtigen Sie die folgenden Probleme, die die Abwärtskompatibilität beeinträchtigen:

- Die Replikationskonfigurations-XML V2 enthält das `Filter`-Element für Regeln. Mit dem `Filter`-Element können Sie Objektfilter basierend auf dem Objektschlüsselpräfix, Tags oder einer Kombination aus beidem angeben, um die Objekte festzulegen, für die die Regel gilt. Die Replikations-Konfiguration XML V1 unterstützt Filterung, die ausschließlich auf dem Schlüsselpräfix basiert. In diesem Fall fügen Sie `Prefix` wie im folgenden Beispiel direkt als untergeordnetes Element des Elements `Rule` hinzu.

  ```
  <?xml version="1.0" encoding="UTF-8"?>
  <ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Role>arn:aws:iam::account-id:role/role-name</Role>
    <Rule>
      <Status>Enabled</Status>
      <Prefix>key-prefix</Prefix>
      <Destination><Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET</Bucket></Destination>

    </Rule>
  </ReplicationConfiguration>
  ```

  Zur Wahrung der Abwärtskompatibilität unterstützt Amazon S3 auch weiterhin die V1-Konfiguration.

- Wenn Sie ein Objekt aus Ihrem Quell-Bucket löschen, ohne eine Objektversions-ID anzugeben, fügt Amazon S3 eine Löschmarkierung hinzu. Wenn Sie V1 der Replikationskonfigurations-XML verwenden, repliziert Amazon S3 Löschmarkierungen, die von Benutzeraktionen herstammen. Anders ausgedrückt: Amazon S3 repliziert die Löschmarkierung nur, wenn ein Benutzer ein Objekt löscht. Wenn ein abgelaufenes Objekt von Amazon S3 (im Rahmen einer Lebenszyklusaktion) entfernt wird, repliziert Amazon S3 die Löschmarkierung nicht.

  In V2-Replikationskonfigurationen können Sie die Replikation von Löschmarkierungen für non-tag-based Regeln aktivieren. Weitere Informationen finden Sie unter Replizieren von Löschmarkierungen auf Buckets.

Einrichten von Berechtigungen

Wenn Sie die Replikation einrichten, müssen Sie die erforderlichen Berechtigungen wie folgt einholen:

- Amazon S3 benötigt Berechtigungen, um Objekte in Ihrem Namen zu replizieren. Sie erteilen diese Berechtigungen, indem Sie eine IAM-Rolle erstellen und die Rolle in der Replikationskonfiguration festlegen.
- Wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener Konten befinden, muss der Eigentümer des Ziel-Buckets dem Quell-Bucket-Eigentümer die Berechtigungen zum Speichern der Replikate erteilen.

Themen

Erstellen einer IAM-Rolle

Standardmäßig sind alle Amazon-S3-Ressourcen – Buckets, Objekte und zugehörige Unterressourcen – privat, sodass nur der Ressourcenbesitzer auf die Ressource zugreifen kann. Amazon S3 benötigt Berechtigungen zum Lesen und Replizieren von Objekten aus dem Quell-Bucket. Sie erteilen diese Berechtigungen, indem Sie eine IAM-Rolle erstellen und die Rolle in der Replikations-Konfiguration festlegen.

In diesem Abschnitt werden die Vertrauensrichtlinie und die mindestens erforderliche Berechtigungsrichtlinie erläutert. Die Beispiel-Walkthroughs enthalten step-by-step Anweisungen zum Erstellen einer IAM-Rolle. Weitere Informationen finden Sie unter Anleitungen: Beispiele zum Konfigurieren der Replikation.

- Im folgenden Beispiel wird eine Vertrauensrichtlinie gezeigt, bei der Sie Amazon S3 als den Service-Prinzipal identifizieren, der die Rolle übernehmen kann.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

- Im folgenden Beispiel wird eine Vertrauensrichtlinie gezeigt, bei der Sie Amazon S3 und S3-Batchvorgänge als Service-Prinzipale identifizieren. Dies ist nützlich, wenn Sie einen Batchreplikationsauftrag erstellen. Weitere Informationen finden Sie unter Erstellen eines Batch-Replikationsauftrags für eine erste Replikationsregel oder ein neues Ziel.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service": [
              "s3.amazonaws.com",
              "batchoperations.s3.amazonaws.com"
           ]
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

  Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

- Im folgenden Beispiel wird eine Zugriffsrichtlinie gezeigt, bei der Sie der Rolle die Berechtigungen erteilen, Replikationsaufgaben in Ihrem Namen durchzuführen. Wenn Amazon S3 die Rolle annimmt, verfügt es über die Berechtigungen, die Sie in dieser Richtlinie angeben. In dieser Richtlinie ist *DOC-EXAMPLE-BUCKET1* der Quell-Bucket und *DOC-EXAMPLE-BUCKET2* der Ziel-Bucket.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl",
            "s3:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*"
      }
   ]
}
```

Die Zugriffsrichtlinie erteilt Berechtigungen für folgenden Aktionen:

- `s3:GetReplicationConfiguration` und `s3:ListBucket` – Berechtigungen für diese Aktionen im *DOC-EXAMPLE-BUCKET1*-Bucket (Quell-Bucket) erlauben es Amazon S3, die Replikationskonfiguration abzurufen und den Bucket-Inhalt aufzulisten. (Das aktuelle

Berechtigungsmodell erfordert die `s3:ListBucket`-Berechtigung für den Zugriff auf Löschmarkierungen.)

- `s3:GetObjectVersionForReplication` und `s3:GetObjectVersionAcl` – Berechtigungen für diese Aktionen, die für alle Objekte erteilt wurden, erlauben es Amazon S3, eine bestimmte Objektversion und eine mit Objekten verknüpfte Zugriffssteuerungsliste (ACL) abzurufen.

- `s3:ReplicateObject` und `s3:ReplicateDelete` – Berechtigungen für diese Aktionen für alle Objekte im *`DOC-EXAMPLE-BUCKET2`*-Bucket (Ziel-Bucket) erlauben es Amazon S3, Objekte oder Löschmarkierungen in den Ziel-Bucket zu replizieren. Informationen zu Löschmarkierungen finden Sie unter Auswirkungen von LöschVorgänge auf die Replikation.

  > **Note**
  >
  > Berechtigungen für die Aktion `s3:ReplicateObject` im *`DOC-EXAMPLE-BUCKET2`*-Bucket (dem Ziel-Bucket) ermöglichen auch die Replikation von Metadaten wie Objekt-Tags und ACLs. Daher müssen Sie für die `s3:ReplicateTags`-Aktion keine explizite Berechtigung erteilen.

- `s3:GetObjectVersionTagging` – Berechtigungen für diese Aktion für Objekte im *`DOC-EXAMPLE-BUCKET1`*-Bucket (Quell-Bucket) gestatten es Amazon S3, Objekt-Tags für die Replikation zu lesen. Weitere Informationen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen. Wenn Amazon S3 nicht über diese Berechtigungen verfügt, repliziert es die Objekte, aber nicht die Objekt-Markierungen.

Eine Liste der Amazon-S3-Aktionen finden Sie unter Amazon S3-Richtlinienaktionen.

> **Important**
>
> Das AWS-Konto , dem die IAM-Rolle gehört, muss über Berechtigungen für die Aktionen verfügen, die es der IAM-Rolle gewährt.
> Angenommen, der Quell-Bucket enthält beispielsweise Objekte, die im Besitz eines anderen AWS-Konto sind. Der Eigentümer der Objekte muss dem AWS-Konto , das die IAM-Rolle besitzt, die erforderlichen Berechtigungen explizit über die Objekt-ACL erteilen. Andernfalls kann Amazon S3 nicht auf die Objekte zugreifen, und die Replikation dieser Objekte schlägt fehl. Weitere Informationen zu ACL-Berechtigungen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.

Die hier beschriebenen Berechtigungen gehören zur Mindest-Replikationskonfiguration. Wenn Sie optionale Replikations-Konfigurationen hinzufügen möchten, müssen Sie Amazon S3 zusätzliche Berechtigungen erteilen. Weitere Informationen finden Sie unter Zusätzliche Replikations-Konfigurationen.

Erteilen von Berechtigungen, wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten

Wenn sich der Quell- und der Ziel-Bucket im Besitz von unterschiedlichen Konten befinden, muss der Eigentümer des Ziel-Buckets auch eine Bucket-Richtlinie hinzufügen, um dem Eigentümer des Quell-Buckets die Berechtigung zum Ausführen von Replikationsaktionen wie folgt zu erteilen. In dieser Richtlinie ist *DOC-EXAMPLE-BUCKET2* der Ziel-Bucket.

Note

Das ARN-Format der Rolle kann anders aussehen. Wenn die Rolle mit der Konsole erstellt wurde, lautet das ARN-Format arn:aws:iam::*account-ID*:role/service-role/*role-name*. Wenn die Rolle mit der erstellt wurde AWS CLI, lautet das ARN-Format arn:aws:iam::*account-ID* :role/*role-name*. Weitere Informationen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

```
{
   "Version":"2012-10-17",
   "Id":"PolicyForDestinationBucket",
   "Statement":[
      {
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
            "s3:ReplicateDelete",
            "s3:ReplicateObject"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*"
      },
```

```
      {
         "Sid":"Permissions on bucket",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action": [
            "s3:List*",
            "s3:GetBucketVersioning",
            "s3:PutBucketVersioning"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET2"
      }
   ]
}
```

Ein Beispiel finden Sie unter Konfigurieren der Replikation, wenn sich Quell- und Ziel-Buckets im Eigentum verschiedener Konten befinden.

Wenn Objekte im Quell-Bucket mit einem Tag versehen sind, beachten Sie Folgendes:

- Wenn der Eigentümer des Quell-Buckets Amazon S3 die Berechtigung für die Aktionen `s3:GetObjectVersionTagging` und `s3:ReplicateTags` zum Replizieren von Objekt-Tags (über die IAM-Rolle) erteilt, repliziert Amazon S3 die Tags zusammen mit den Objekten. Weitere Information zur IAM-Rolle finden Sie unter Erstellen einer IAM-Rolle.
- Wenn der Eigentümer des Ziel-Buckets die Tags nicht replizieren will, kann er die folgende Anweisung zur Richtlinie für den Ziel-Bucket hinzufügen, um die Berechtigung für die Aktion `s3:ReplicateTags` explizit zu verweigern. In dieser Richtlinie ist *DOC-EXAMPLE-BUCKET2* der Ziel-Bucket.

```
...
   "Statement":[
      {
         "Effect":"Deny",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-id:role/service-role/source-
account-IAM-role"
         },
         "Action":"s3:ReplicateTags",
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*"
      }
```

```
    ]
...
```

Gewähren von Berechtigungen für S3-Batch-Operationen

Die S3-Batch-Replikation bietet Ihnen eine Möglichkeit, Objekte zu replizieren, die existierten, bevor eine Replikationskonfiguration vorhanden war, Objekte, die zuvor repliziert wurden, und Objekte, bei denen die Replikation fehlgeschlagen ist. Beim Erstellen der ersten Regel in einer neuen Replikationskonfiguration oder beim Hinzufügen eines neuen Ziels zu einer vorhandenen Konfiguration über die AWS Management Console können Sie einen einmaligen Batch-Replikationsauftrag erstellen. Sie können die Batch-Replikation auch für eine bestehende Replikationskonfiguration initiieren, indem Sie einen Batch-Operationsauftrag erstellen.

Beispiele für Batch-Replikations-IAM-Rollen und -Richtlinien finden Sie unter Konfigurieren von IAM-Richtlinien für die Batch-Replikation.

Ändern des Replikatbesitzers

Wenn der Quell- und der Ziel-Bucket unterschiedlich AWS-Konten sind, können Sie Amazon S3 anweisen, die Eigentümerschaft des Replikats zu dem zu ändern AWS-Konto , dem der Ziel-Bucket gehört. Weitere Informationen zum Außerkraftsetzen des Besitzers finden Sie unter Ändern des Replikat-Eigentümers.

Aktivieren des Empfangs replizierter Objekte aus einem Quell-Bucket

Sie können schnell die Richtlinien generieren, die erforderlich sind, um den Empfang replizierter Objekte aus einem Quell-Bucket über die AWS Management Console zu aktivieren.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Bucket aus, den Sie als Ziel-Bucket verwenden möchten.
4. Wählen Sie die Registerkarte Management (Verwaltung) aus und scrollen Sie nach unten zu Replication rules (Replikationsregeln).
5. Wählen Sie für Actions (Aktionen) die Option Receive replicated objects (Replizierte Objekte empfangen) aus.

Folgen Sie den Anweisungen, geben Sie die AWS-Konto ID des Quell-Bucket-Kontos ein und wählen Sie Richtlinien generieren aus. Daraufhin werden eine Amazon-S3-Bucket-Richtlinie und eine KMS-Schlüsselrichtlinie erstellt.

6. Wenn Sie diese Richtlinie Ihrer bestehenden Bucket-Richtlinie hinzufügen möchten, wählen Sie entweder Apply settings (Einstellungen anwenden) oder Copy (Kopieren) aus, um die Änderungen manuell zu kopieren.
7. (Optional) Kopieren Sie die AWS KMS Richtlinie in die gewünschte KMS-Schlüsselrichtlinie in der AWS Key Management Service -Konsole.

Anleitungen: Beispiele zum Konfigurieren der Replikation

Das folgende Beispiel zeigt das Konfigurieren der Live-Replikation für häufige Anwendungsfälle. Die Beispiele zeigen die Replikationskonfiguration mit der Amazon S3-Konsole, AWS Command Line Interface (AWS CLI) und AWS SDKs (Beispiele für Java- und .NET-SDK werden gezeigt). Informationen zum Installieren und Konfigurieren der AWS CLI finden Sie in den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch.

> **Note**
>
> Die Live-Replikation bezieht sich auf Replikation innerhalb derselben Region (SRR) und die Replikation in mehreren Regionen (CRR). Informationen zu einer On-Demand-Replikationsaktion zum Synchronisieren von Buckets und zum Replizieren vorhandener Objekte finden Sie unter Replizieren vorhandener Objekte.

- Installieren der AWS Command Line Interface
- Konfigurieren der AWS CLI – Sie müssen mindestens ein Profil einrichten. Richten Sie bei kontoübergreifenden Szenarien zwei Profile ein.

Weitere Informationen zu - AWS SDKs finden Sie unter AWS -SDK für Java und AWS -SDK für .NET.

Weitere Informationen zur Verwendung der S3-Replikation zum Replizieren von Daten finden Sie unter Tutorial: Replizieren von Daten innerhalb und zwischen AWS-Regionen mithilfe der S3-Replikation.

Themen

- Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos
- Konfigurieren der Replikation, wenn sich Quell- und Ziel-Buckets im Eigentum verschiedener Konten befinden
- Ändern des Replikat-Eigentümers, wenn sich Quell- und Ziel-Buckets im Eigentum unterschiedlicher Konten befinden
- Replizieren verschlüsselter Objekte
- Replizieren von Objekten mit S3-Replikationszeitkontrolle (S3 RTC)
- Verwalten von Replikationsregeln mit der Amazon-S3-Konsole

Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos

Die Replikation ist das automatische, asynchrone Kopieren von Objekten über Buckets hinweg in derselben oder einer anderen AWS-Regionen. Sie kopiert neu erstellte Objekte und Objektaktualisierungen aus einem Quell-Bucket in einen Ziel-Bucket oder mehrere Ziel-Buckets. Weitere Informationen finden Sie unter Replizieren von Objekten.

Wenn Sie die Replikation konfigurieren, fügen Sie dem Quell-Bucket Replikationsregeln hinzu. Replikationsregeln definieren, welche Quell-Bucket-Objekte repliziert werden sollen, und den Ziel-Bucket/die Ziel-Buckets, in dem/denen die replizierten Objekte gespeichert werden sollen. Sie können eine Regel erstellen, um alle Objekte in einem Bucket oder eine Untermenge von Objekten mit einem spezifischen Schlüsselnamenpräfixen, einem oder mehreren Objekt-Markierungen oder beidem zu replizieren. Ein Ziel-Bucket kann sich in derselben AWS-Konto wie der Quell-Bucket oder in einem anderen Konto befinden.

Wenn Sie angeben, dass eine Objektversions-ID gelöscht werden soll, löscht Amazon S3 diese Objektversion im Quell-Bucket. Es repliziert die Löschung aber nicht im Ziel-Bucket. Anders ausgedrückt: Dieselbe Objektversion wird im Ziel-Bucket nicht gelöscht. Dies schützt Daten vor missbräuchlichen Löschungen.

Wenn Sie einem Bucket eine Replikationsregel hinzufügen, ist diese standardmäßig aktiviert, sodass sie ausgeführt wird, sobald Sie sie speichern.

In diesem Beispiel richten Sie eine Replikation für Quell- und Ziel-Buckets ein, die demselben AWS-Konto gehören. Beispiele für die Verwendung der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI) und der AWS SDK for Java und AWS SDK for .NET.

Verwenden der S3-Konsole

Gehen Sie folgendermaßen vor, um eine Replikationsregel zu konfigurieren, wenn sich der Ziel-Bucket in derselben AWS-Konto wie der Quell-Bucket befindet.

Wenn sich der Ziel-Bucket in einem anderen Konto als der Quell-Bucket befindet, müssen Sie eine Bucket-Richtlinie für den Ziel-Bucket hinzufügen, um dem Eigentümer des Quell-Bucket-Kontos die Berechtigung zum Replizieren von Objekten in der Ziel-Bucket zu erteilen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des von Ihnen erstellten Buckets aus.
4. Wählen Sie die Registerkarte Verwaltung aus, scrollen Sie nach unten zu Replikationsregeln und wählen Sie dann Replikationsregel erstellen aus.
5. Geben Sie im Abschnitt Replikationsregelkonfiguration unter Name der Replikationsregel einen Namen für Ihre Regel ein, um die Regel später leichter identifizieren zu können. Der Name ist erforderlich und muss innerhalb des Buckets eindeutig sein.
6. In Status (Status)ist Enabled (Aktiviert) standardmäßig ausgewählt. Eine aktivierte Regel wird ausgeführt, sobald Sie speichern. Wenn Sie die Regel später aktivieren möchten, wählen Sie Deaktiviert aus.
7. Wenn es bereits Replikationsregeln für den Bucket gibt, werden Sie angewiesen, eine Priorität für die Regel festzulegen. Sie müssen eine Priorität für die Regel festlegen, um Konflikte zu vermeiden, die durch Objekte verursacht werden, die mehreren Regeln unterliegen. Wenn sich Regeln überschneiden, verwendet Amazon S3 die Regelpriorität, um die Regel zu ermitteln, die angewendet werden muss. Je höher die Zahl, desto höher die Priorität. Weitere Informationen zur Priorität von Regeln finden Sie unter Replikations-Konfiguration.
8. Unter Quell-Bucket stehen Ihnen folgende Optionen zum Festlegen der Replikationsquelle zur Verfügung:

   - Um den gesamten Bucket zu replizieren, wählen Sie Apply to all objects in the bucket (Auf alle Objekte im Bucket anwenden).
   - Um alle Objekte zu replizieren, die dasselbe Präfix haben, wählen Sie Limit the scope of this rule using one or more filters (Geltungsbereich dieser Regel mit einem oder mehreren Filtern

einschränken). Dies beschränkt die Replikation auf alle Objekte, deren Namen mit dem von Ihnen angegebenen Präfix beginnen (z. B. `pictures`). Geben Sie ein Präfix in das Feld Präfix ein.

> **Note**
>
> Wenn Sie ein Präfix eingeben, das den Namen eines Ordners darstellt, müssen Sie / (Schrägstrich) als letztes Zeichen eingeben (z. B. `pictures/`).

- Um alle Objekte mit einem oder mehreren Objekt-Tags zu replizieren, wählen Sie Tag hinzufügen aus und geben Sie das Schlüssel-Wert-Paar in die Felder ein. Wiederholen Sie den Vorgang, um ein weiteres Tag hinzuzufügen. Sie können ein Präfix und Markierungen kombinieren. Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

Das neue XML-Schema für die Replikationskonfiguration unterstützt das Filtern nach Präfix und Tag und die Priorisierung von Regeln. Weitere Informationen über das neue Schema finden Sie unter Abwärtskompatibilität. Weitere Informationen über das mit der Amazon-S3-API verwendete XML, das hinter der Benutzeroberfläche funktioniert, finden Sie unter Replikations-Konfiguration. Das neue Schema wird als XML V2 für die Replikationskonfiguration beschrieben.

9. Wählen Sie unter Ziel den Bucket aus, in den Amazon S3 Objekte replizieren soll.

> **Note**
>
> Die Anzahl der Ziel-Buckets ist auf die Anzahl der AWS-Regionen in einer bestimmten Partition begrenzt. Eine Partition ist eine Gruppierung von Regionen. hat AWS derzeit drei Partitionen: `aws` (Standardregionen), `aws-cn` (China-Regionen) und `aws-us-gov` (AWS GovCloud (US) Regionen). Sie können Service Quotas verwenden, um eine Erhöhung Ihres Ziel-Bucket-Kontingents anzufordern.

- Um in einen Bucket oder mehrere Buckets in Ihrem Konto zu replizieren, wählen Sie Bucket in diesem Konto wählen aus und geben Sie die Ziel-Bucket-Namen ein oder suchen Sie danach.
- Um in einen Bucket oder mehrere Buckets in einem anderen zu replizieren AWS-Konto, wählen Sie Bucket in einem anderen Konto angeben und geben Sie die Konto-ID des Ziel-Buckets und den Bucket-Namen ein.

Wenn sich das Ziel in einem anderen Konto als der Quell-Bucket befindet, müssen Sie eine Bucket-Richtlinie für die Ziel-Buckets hinzufügen, um dem Eigentümer des Quell-Bucket-Kontos die Berechtigung zum Replizieren von Objekten zu erteilen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten.

Wenn Sie eine leichtere Standardisierung der Eigentümerschaft für neue Objekte im Ziel-Bucket ermöglichen möchten, wählen Sie Objekteigentümerschaft in Eigentümer des Ziel-Buckets ändern aus. Weitere Informationen zu dieser Option finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

> **Note**
>
> Wenn die Versionssteuerung für den Ziel-Bucket nicht aktiviert ist, erhalten Sie eine Warnmeldung, die die Schaltfläche Versionierung aktivieren enthält. Wählen Sie diese Schaltfläche, um das Versioning für den Bucket zu aktivieren.

10. Richten Sie eine AWS Identity and Access Management (IAM)-Rolle ein, die Amazon S3 annehmen kann, um Objekte in Ihrem Namen zu replizieren.

    Um eine IAM-Rolle einzurichten, wählen Sie im Abschnitt IAM-Rolle eine der folgenden Optionen aus der Dropdown-Liste IAM-Rolle aus:

    - Wir empfehlen Ihnen, Create new role (Neue Rolle erstellen) auszuwählen, um Amazon S3 anzuweisen, eine neue IAM-Rolle für Sie zu erstellen. Wenn Sie die Regel speichern, wird eine neue Richtlinie für die IAM-Rolle erstellt, die mit den von Ihnen ausgewählten Quell- und Ziel-Buckets übereinstimmt.
    - Sie haben auch die Möglichkeit eine vorhandene IAM-Rolle zu verwenden. In diesem Fall müssen Sie eine Rolle auswählen, die Amazon S3 die erforderlichen Berechtigungen für die Replikation gewährt. Die Replikation schlägt fehl, wenn diese Rolle Amazon S3 keine ausreichenden Berechtigungen gewährt, um Ihre Replikationsregel zu befolgen.

> **Important**
>
> Wenn Sie eine Replikationsregel zu einem Bucket hinzufügen, benötigen Sie die Berechtigung `iam:PassRole` zum Übergeben der IAM-Rolle, die Amazon S3 Replikationsberechtigungen erteilt. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Serviceübergeben kann im IAM-Benutzerhandbuch.

11. Um Objekte im Quell-Bucket zu replizieren, die mit serverseitiger Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) verschlüsselt sind, wählen Sie unter Verschlüsselung die Option Mit verschlüsselte Objekte replizieren aus AWS KMS. Unter AWS KMS -Schlüssel für die Verschlüsselung von Zielobjekten befinden sich die Quellschlüssel, deren Verwendung Sie der Replikation erlauben. Alle Quell-KMS-Schlüssel sind standardmäßig enthalten. Um die KMS-Schlüsselauswahl einzuschränken, können Sie einen Alias oder eine Schlüssel-ID auswählen.

    Mit verschlüsselte Objekte AWS KMS keys , die Sie nicht auswählen, werden nicht repliziert. Ein KMS-Schlüssel oder eine Gruppe von KMS-Schlüsseln wird für Sie ausgewählt, aber Sie können die KMS-Schlüssel auch selbst auswählen. Informationen zur Verwendung von AWS KMS mit der Replikation finden Sie unter Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten.

    > **Important**
    >
    > Wenn Sie Objekte replizieren, die mit verschlüsselt sind AWS KMS, verdoppelt sich die AWS KMS Anforderungsrate in der Quellregion und erhöht sich in der Zielregion um den gleichen Betrag. Diese erhöhten Aufrufraten für AWS KMS sind auf die Art und Weise zurückzuführen, wie Daten mithilfe des KMS-Schlüssels erneut verschlüsselt werden, den Sie für die Zielregion der Replikation definieren. AWS KMS hat ein Anforderungsratenkontingent, das pro aufrufendem Konto und Region gilt. Informationen zu den standardmäßigen Kontingenten finden Sie unter AWS KMS - Kontingente – Anforderungen pro Sekunde: variabel im AWS Key Management Service Entwicklerhandbuch.
    > Wenn Ihre aktuelle Amazon S3-PUTObjektanforderungsrate während der Replikation mehr als die Hälfte des AWS KMS Standardratenlimits für Ihr Konto beträgt, empfehlen wir Ihnen, eine Erhöhung Ihres AWS KMS Anforderungsratenkontingents anzufordern.

Wenn Sie eine Erhöhung anfordern möchten, erstellen Sie einen Fall im AWS Support -Center unter Contact Us (Kontakt). Angenommen, Ihre aktuelle PUT Objektanforderungsrate beträgt 1 000 Anforderungen pro Sekunde und Sie verwenden , AWS KMS um Ihre Objekte zu verschlüsseln. In diesem Fall empfehlen wir Ihnen, AWS Support zu bitten, Ihr AWS KMS Ratenlimit sowohl in Ihrer Quell- als auch in Ihrer Zielregion (falls unterschiedlich) auf 2 500 Anforderungen pro Sekunde zu erhöhen, um sicherzustellen, dass es keine Drosselung durch gibt AWS KMS.
Um Ihre PUT Objektanforderungsrate im Quell-Bucket anzuzeigen, sehen Sie sich `PutRequests` in den Amazon- CloudWatch Anforderungsmetriken für Amazon S3 an. Weitere Informationen zum Anzeigen von CloudWatch Metriken finden Sie unter Verwenden der S3-Konsole.

Wenn Sie mit verschlüsselte Objekte replizieren möchten AWS KMS, gehen Sie wie folgt vor:

- Geben Sie unter AWS KMS key für die Verschlüsselung von Zielobjekten Ihren KMS-Schlüssel auf eine der folgenden Arten an:
  - Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS keys wählen und anschließend den KMS-Schlüssel in der Liste der verfügbaren Schlüssel aus.

    Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS -Schlüssel im Entwicklerhandbuch zu AWS Key Management Service .
  - Wählen Sie zum Eingeben des Amazon-Ressourcennamens (ARN) des KMS-Schlüssels AWS KMS key -ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein. Dadurch werden die Replikate im Ziel-Bucket verschlüsselt. Sie finden den ARN für Ihren KMS-Schlüssel in der IAM-Konsole unter Verschlüsselungsschlüssel.
  - Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

    Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

> **Important**
>
> Sie können nur KMS-Schlüssel verwenden, die in derselben AWS-Region wie der Bucket aktiviert sind. Wenn Sie Aus Ihren KMS-Schlüsseln wählen auswählen, werden in der S3-Konsole nur 100 KMS-Schlüssel pro Region aufgeführt. Wenn Sie über mehr als 100 KMS-Schlüssel in derselben Region verfügen, können Sie nur die ersten 100 KMS-Schlüssel in der S3-Konsole sehen. Um einen nicht in der Konsole aufgeführten KMS-Schlüssel zu verwenden, wählen Sie AWS KMS key - ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN ein.
> Wenn Sie einen AWS KMS key für die serverseitige Verschlüsselung in Amazon S3 verwenden, müssen Sie einen KMS-Schlüssel mit symmetrischer Verschlüsselung auswählen. Amazon S3 unterstützt nur KMS-Schlüssel mit symmetrischer Verschlüsselung und keine asymmetrischen KMS-Schlüssel. Weitere Informationen finden Sie unter Erkennen von symmetrischen und asymmetrischen KMS-Schlüsseln im Entwicklerhandbuch zu AWS Key Management Service .

Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch. Weitere Informationen zur Verwendung von AWS KMS mit Amazon S3 finden Sie unter Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

12. Wenn Sie Ihre Daten in eine bestimmte Speicherklasse im Ziel replizieren wollen, wählen Sie unter Zielspeicherklasse die Option Speicherklasse für die replizierten Objekte ändern aus. Anschließend wählen Sie die Speicherklasse, die Sie für die replizierten Objekte im Ziel verwenden wollen. Wenn Sie diese Option nicht auswählen, ist die Speicherklasse für replizierte Objekte dieselbe Klasse wie für die ursprünglichen Objekte.
13. Beim Festlegen der zusätzlichen Replikationsoptionen haben Sie folgende zusätzliche Optionen:

    - Wenn Sie die Begrenzung der S3-Replikationszeit (S3 RTC) in Ihrer Replikationskonfiguration aktivieren möchten, wählen Sie Replikationszeitkontrolle (Replication Time Control, RTC) aus. Weitere Informationen zu dieser Option finden Sie unter Erfüllen der Compliance-Anforderungen mit S3-Replikationszeitkontrolle (S3 RTC).
    - Wenn Sie S3-Replikationsmetriken in Ihrer Replikationskonfiguration aktivieren möchten, wählen Sie Replication metrics and events (Replikationsmetriken und -ereignisse) aus. Weitere

Informationen finden Sie unter Überwachen des Fortschritts mit Replikationsmetriken und S3-Ereignisbenachrichtigungen.

- Wenn Sie die Löschmarkierungs-Replikation in Ihrer Replikations-Konfiguration aktivieren möchten, wählen Sie Markierungsreplikation löschen aus. Weitere Informationen finden Sie unter Replizieren von Löschmarkierungen auf Buckets.
- Wenn Sie die Synchronisierung der Amazon-S3-Replikatänderung in Ihrer Replikations-Konfiguration aktivieren möchten, wählen Sie Synchronisierung der Replikatänderung aus. Weitere Informationen finden Sie unter Replizieren von Metadatenänderungen mit der Synchronisierung von Amazon-S3-Replikatänderungen.

> **Note**
>
> Wenn Sie S3-RTC- oder S3-Replikationsmetriken verwenden, fallen zusätzliche Gebühren an.

14. Zum Abschluss wählen Sie Save (Speichern).
15. Nachdem Sie Ihre Regel gespeichert haben, können Sie Ihre Regel bearbeiten, aktivieren, deaktivieren oder löschen, indem Sie Ihre Regel auswählen und Edit rule (Regel bearbeiten) wählen.

Verwenden der AWS CLI

Gehen Sie wie folgt vor, um die AWS CLI zum Einrichten der Replikation zu verwenden AWS-Konto, wenn sich Quell- und Ziel-Bucket im Besitz desselben befinden:

- Erstellen Sie Quell- und Ziel-Buckets
- Aktivieren Sie das Versioning für die Buckets
- Erstellen Sie eine IAM-Rolle welche Amazon S3 die Berechtigungen erteilt, Objekte zu replizieren.
- Fügen Sie die Replikationskonfiguration zum Quell-Bucket hinzu

Um die Einrichtung zu prüfen, testen Sie sie.

So richten Sie die Replikation ein, wenn sich Quell- und Ziel-Bucket im Besitz desselben befinden AWS-Konto

1. Richten Sie das Anmeldeinformationsprofil für die AWS CLI ein. In diesem Beispiel verwenden wir den Profilnamen `acctA`. Informationen zum Einrichten der Anmeldeinformations-Profile finden Sie unter Named Profiles (Benannte Profile) im AWS Command Line Interface - Benutzerhandbuch.

   > **Important**
   >
   > Das Profil, das Sie für diese Übung verwenden, muss über die nötigen Berechtigungen verfügen. Beispielsweise legen Sie in der Replikations-Konfiguration die IAM-Rolle fest, die Amazon S3 annehmen kann. Dies können Sie nur tun, wenn das verwendete Profil über die `iam:PassRole`-Berechtigung verfügt. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS -Service übergeben kann im IAM-Benutzerhandbuch. Wenn Sie zur Erstellung eines benannten Profils die Anmeldeinformationen eines Administrators verwenden, können Sie alle Aufgaben durchführen.

2. Erstellen Sie einen *`source`*-Bucket und aktivieren Sie dafür das Versioning. Der folgende Code erstellt einen *`source`*-Bucket in der Region USA Ost (Nord-Virginia) (us-east-1).

   ```
   aws s3api create-bucket \
   --bucket source \
   --region us-east-1 \
   --profile acctA
   ```

   ```
   aws s3api put-bucket-versioning \
   --bucket source \
   --versioning-configuration Status=Enabled \
   --profile acctA
   ```

3. Erstellen Sie einen *`destination`*-Bucket und aktivieren Sie dafür das Versioning. Der folgende Code erstellt einen *`destination`*-Bucket in der Region US West (Oregon) (us-west-2).

> **Note**
>
> Um die Replikationskonfiguration einzurichten, wenn sich Quell- und Ziel-Bucket im selben befinden AWS-Konto, verwenden Sie dasselbe Profil. Dieses Beispiel verwendet `acctA`. Um die Replikationskonfiguration zu testen, wenn sich die Buckets im Besitz verschiedener befinden AWS-Konten, geben Sie für jede unterschiedliche Profile an. In diesem Beispiel verwenden wir das Profil `acctB` für den Ziel-Bucket.

```
aws s3api create-bucket \
--bucket destination \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket destination \
--versioning-configuration Status=Enabled \
--profile acctA
```

4. Erstellen Sie eine IAM-Rolle. Sie geben diese Rolle in der Replizierungskonfiguration an, die Sie später zum *Quell-Bucket* hinzufügen. Amazon S3 übernimmt diese Rolle, um Objekte in Ihrem Namen zu replizieren. Sie erstellen eine IAM-Rolle in zwei Schritten:

   - Erstellen Sie eine Rolle.
   - Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

   a. Erstellen Sie die IAM-Rolle.

      i. Kopieren Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-trust-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie gewährt Amazon S3 Service-Prinzipal-Berechtigungen, um die Rolle anzunehmen.

```
{
   "Version":"2012-10-17",
   "Statement":[
```

```
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

ii. Führen Sie den folgenden Befehl aus, um eine Rolle zu erstellen.

```
$ aws iam create-role \
--role-name replicationRole \
--assume-role-policy-document file://s3-role-trust-policy.json  \
--profile acctA
```

b. Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

i. Kopieren Sie die folgende Berechtigungsrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-permissions-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Zugriffsrichtlinie erteilt Berechtigungen für verschiedene Amazon-S3-Bucket- und -Objektaktionen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl",
            "s3:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration"
```

```
            ],
            "Resource":[
               "arn:aws:s3:::source-bucket"
            ]
         },
         {
            "Effect":"Allow",
            "Action":[
               "s3:ReplicateObject",
               "s3:ReplicateDelete",
               "s3:ReplicateTags"
            ],
            "Resource":"arn:aws:s3:::destination-bucket/*"
         }
      ]
}
```

ii. Führen Sie den folgenden Befehl aus, um eine Richtlinie zu erstellen und sie der Rolle anzufügen.

```
$ aws iam put-role-policy \
--role-name replicationRole \
--policy-document file://s3-role-permissions-policy.json \
--policy-name replicationRolePolicy \
--profile acctA
```

5. Fügen Sie eine Replikationskonfiguration zum *source*-Bucket hinzu.

   a. Obwohl die Amazon S3-API eine Replikationskonfiguration als XML erfordert, AWS CLI erfordert die , dass Sie die Replikationskonfiguration als JSON angeben. Speichern Sie den folgenden JSON-Code in einer Datei mit dem Namen `replication.json` im lokalen Verzeichnis auf Ihrem Computer.

```
{
  "Role": "IAM-role-ARN",
  "Rules": [
    {
      "Status": "Enabled",
      "Priority": 1,
      "DeleteMarkerReplication": { "Status": "Disabled" },
      "Filter" : { "Prefix": "Tax"},
      "Destination": {
        "Bucket": "arn:aws:s3:::destination-bucket"
```

```
      }
    }
  ]
}
```

b. Aktualisieren Sie den JSON-Code, indem Sie Werte für die *destination-bucket* und *IAM-role-ARN* angeben. Speichern Sie die Änderungen.

c. Führen Sie den folgenden Befehl aus, um die Replikations-Konfiguration zu Ihrem Quell-Bucket hinzuzufügen. Stellen Sie sicher, dass Sie den Namen für den *source*-Bucket angeben.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket source \
--profile acctA
```

Um die Replikationskonfiguration abzurufen, verwenden Sie den Befehl `get-bucket-replication`.

```
$ aws s3api get-bucket-replication \
--bucket source \
--profile acctA
```

6. Testen Sie das Setup in der Amazon-S3-Konsole:

a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

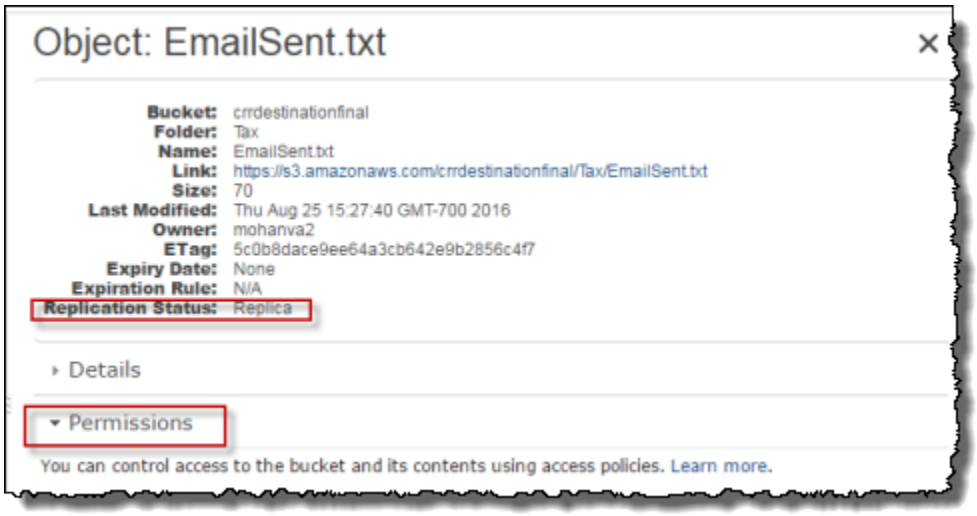
b. Erstellen Sie im *source*-Bucket einen Ordner mit dem Namen `Tax`.

c. Fügen Sie Beispielobjekte zum `Tax`-Ordner im *source*-Bucket hinzu.

> **Note**
>
> Die von Amazon S3 für die Replikation eines Objekts benötigte Zeit hängt von der Größe des Objekts ab. Weitere Informationen zum Anzeigen des Replikationsstatus finden Sie unter Abrufen von Replikationsstatusinformationen.

Überprüfen Sie im *destination*-Bucket Folgendes:

- Dass Amazon S3 die Objekte repliziert hat.
- In den properties (Eigenschaften) des Objekts, dass der Replication Status (Replikationsstatus) auf `Replica` festgelegt ist (was es als Replikatobjekt kennzeichnet).
- In den properties (Eigenschaften) des Objekts, dass im Berechtigungsabschnitt keine Berechtigungen aufgeführt sind. Dies bedeutet, dass sich das Replikat noch im Besitz des *`source`*-Bucket-Eigentümers befindet und der *`destination`*-Bucket-Eigentümer über keine Berechtigung auf dem Objektreplikat verfügt. Sie können eine optionale Konfiguration hinzufügen, um Amazon S3 anzuweisen, die Replikat-Eigentümerschaft zu ändern. Ein Beispiel finden Sie unter Ändern des Replikat-Eigentümers, wenn sich Quell- und Ziel-Buckets im Eigentum unterschiedlicher Konten befinden.



Verwenden der AWS SDKs

Verwenden Sie die folgenden Codebeispiele, um einem Bucket mit bzw. eine Replikationskonfiguration hinzuzufügen AWS SDK for Java AWS SDK for .NET.

Java

Das folgende Beispiel fügt eine Replikations-Konfiguration einem Bucket hinzu und ruft die Konfiguration anschließend ab und überprüft sie. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
```

```
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.identitymanagement.AmazonIdentityManagement;
import
 com.amazonaws.services.identitymanagement.AmazonIdentityManagementClientBuilder;
import com.amazonaws.services.identitymanagement.model.CreateRoleRequest;
import com.amazonaws.services.identitymanagement.model.PutRolePolicyRequest;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.BucketReplicationConfiguration;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.CreateBucketRequest;
import com.amazonaws.services.s3.model.DeleteMarkerReplication;
import com.amazonaws.services.s3.model.DeleteMarkerReplicationStatus;
import com.amazonaws.services.s3.model.ReplicationDestinationConfig;
import com.amazonaws.services.s3.model.ReplicationRule;
import com.amazonaws.services.s3.model.ReplicationRuleStatus;
import com.amazonaws.services.s3.model.SetBucketVersioningConfigurationRequest;
import com.amazonaws.services.s3.model.StorageClass;
import com.amazonaws.services.s3.model.replication.ReplicationFilter;
import com.amazonaws.services.s3.model.replication.ReplicationFilterPredicate;
import com.amazonaws.services.s3.model.replication.ReplicationPrefixPredicate;

import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class CrossRegionReplication {

        public static void main(String[] args) throws IOException {
                Regions clientRegion = Regions.DEFAULT_REGION;
                String accountId = "*** Account ID ***";
                String roleName = "*** Role name ***";
                String sourceBucketName = "*** Source bucket name ***";
                String destBucketName = "*** Destination bucket name ***";
                String prefix = "Tax/";

                String roleARN = String.format("arn:aws:iam::%s:%s", accountId,
 roleName);
                String destinationBucketARN = "arn:aws:s3:::" + destBucketName;
```

```
                AmazonS3 s3Client = AmazonS3Client.builder()
                                .withCredentials(new ProfileCredentialsProvider())
                                .withRegion(clientRegion)
                                .build();

                createBucket(s3Client, clientRegion, sourceBucketName);
                createBucket(s3Client, clientRegion, destBucketName);
                assignRole(roleName, clientRegion, sourceBucketName,
destBucketName);

                try {

                        // Create the replication rule.
                        List<ReplicationFilterPredicate> andOperands = new
ArrayList<ReplicationFilterPredicate>();
                        andOperands.add(new ReplicationPrefixPredicate(prefix));

                        Map<String, ReplicationRule> replicationRules = new
HashMap<String, ReplicationRule>();
                        replicationRules.put("ReplicationRule1",
                                        new ReplicationRule()
                                                        .withPriority(0)

.withStatus(ReplicationRuleStatus.Enabled)

.withDeleteMarkerReplication(
                                                                        new

DeleteMarkerReplication().withStatus(

   DeleteMarkerReplicationStatus.DISABLED))
                                                        .withFilter(new

ReplicationFilter().withPredicate(
                                                                        new

ReplicationPrefixPredicate(prefix)))
                                                        .withDestinationConfig(new

ReplicationDestinationConfig()

.withBucketARN(destinationBucketARN)

.withStorageClass(StorageClass.Standard)));

                        // Save the replication rule to the source bucket.
                        s3Client.setBucketReplicationConfiguration(sourceBucketName,
                                        new BucketReplicationConfiguration()
```

```
                                                        .withRoleARN(roleARN)

.withRules(replicationRules));

                        // Retrieve the replication configuration and verify that
the configuration
                        // matches the rule we just set.
                        BucketReplicationConfiguration replicationConfig = s3Client

.getBucketReplicationConfiguration(sourceBucketName);
                        ReplicationRule rule =
replicationConfig.getRule("ReplicationRule1");
                        System.out.println("Retrieved destination bucket ARN: "
                                           +
rule.getDestinationConfig().getBucketARN());
                        System.out.println("Retrieved priority: " +
rule.getPriority());
                        System.out.println("Retrieved source-bucket replication rule
status: " + rule.getStatus());
                } catch (AmazonServiceException e) {
                        // The call was transmitted successfully, but Amazon S3
couldn't process
                        // it, so it returned an error response.
                        e.printStackTrace();
                } catch (SdkClientException e) {
                        // Amazon S3 couldn't be contacted for a response, or the
client
                        // couldn't parse the response from Amazon S3.
                        e.printStackTrace();
                }
        }

        private static void createBucket(AmazonS3 s3Client, Regions region, String
bucketName) {
                CreateBucketRequest request = new CreateBucketRequest(bucketName,
region.getName());
                s3Client.createBucket(request);
                BucketVersioningConfiguration configuration = new
BucketVersioningConfiguration()
                                .withStatus(BucketVersioningConfiguration.ENABLED);

                SetBucketVersioningConfigurationRequest enableVersioningRequest =
new SetBucketVersioningConfigurationRequest(
                                bucketName, configuration);
```

```
                s3Client.setBucketVersioningConfiguration(enableVersioningRequest);

        }

        private static void assignRole(String roleName, Regions region, String
 sourceBucket, String destinationBucket) {
                AmazonIdentityManagement iamClient =
 AmazonIdentityManagementClientBuilder.standard()
                                .withRegion(region)
                                .withCredentials(new ProfileCredentialsProvider())
                                .build();
                StringBuilder trustPolicy = new StringBuilder();
                trustPolicy.append("{\\r\\n   ");
                trustPolicy.append("\\\"Version\\\":\\\"2012-10-17\\\",\\r\\n   ");
                trustPolicy.append("\\\"Statement\\\":[\\r\\n      {\\r\\n
 ");
                trustPolicy.append("\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\
\"Principal\\\":{\\r\\n            ");
                trustPolicy.append("\\\"Service\\\":\\\"s3.amazonaws.com\\\"\\r\\n
       },\\r\\n         ");
                trustPolicy.append("\\\"Action\\\":\\\"sts:AssumeRole\\\"\\r\\n
  }\\r\\n   ]\\r\\n}");

                CreateRoleRequest createRoleRequest = new CreateRoleRequest()
                                .withRoleName(roleName)

 .withAssumeRolePolicyDocument(trustPolicy.toString());

                iamClient.createRole(createRoleRequest);

                StringBuilder permissionPolicy = new StringBuilder();
                permissionPolicy.append(
                                "{\\r\\n   \\\"Version\\\":\\\"2012-10-17\\\",\\r\\n
   \\\"Statement\\\":[\\r\\n      {\\r\\n         ");
                permissionPolicy.append(
                                "\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\
\"Action\\\":[\\r\\n             ");
                permissionPolicy.append("\\\"s3:GetObjectVersionForReplication\\\",\
\r\\n            ");
                permissionPolicy.append(
                                "\\\"s3:GetObjectVersionAcl\\\"\\r\\n         ],\\r\
\n         \\\"Resource\\\":[\\r\\n            ");
                permissionPolicy.append("\\\"arn:aws:s3:::");
                permissionPolicy.append(sourceBucket);
```

```
                permissionPolicy.append("/*\\\"\\r\\n         ]\\r\\n      },\\r\\n
     {\\r\\n         ");
                permissionPolicy.append(
                                "\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\
\"Action\\\":[\\r\\n            ");
                permissionPolicy.append(
                                "\\\"s3:ListBucket\\\",\\r\\n            \\
\"s3:GetReplicationConfiguration\\\"\\r\\n         ");
                permissionPolicy.append("],\\r\\n         \\\"Resource\\\":[\\r\\n
          \\\"arn:aws:s3:::");
                permissionPolicy.append(sourceBucket);
                permissionPolicy.append("\\r\\n         ");
                permissionPolicy
                                .append("]\\r\\n      },\\r\\n      {\\r\\n
 \\\"Effect\\\":\\\"Allow\\\",\\r\\n         ");
                permissionPolicy.append(
                                "\\\"Action\\\":[\\r\\n            \\
\"s3:ReplicateObject\\\",\\r\\n            ");
                permissionPolicy
                                .append("\\\"s3:ReplicateDelete\\\",\\r\\n
  \\\"s3:ReplicateTags\\\",\\r\\n            ");
                permissionPolicy.append("\\\"s3:GetObjectVersionTagging\\\"\\r\\n\\r
\\n         ],\\r\\n         ");
                permissionPolicy.append("\\\"Resource\\\":\\\"arn:aws:s3:::");
                permissionPolicy.append(destinationBucket);
                permissionPolicy.append("/*\\\"\\r\\n      }\\r\\n   ]\\r\\n}");

                PutRolePolicyRequest putRolePolicyRequest = new
 PutRolePolicyRequest()
                                .withRoleName(roleName)
                                .withPolicyDocument(permissionPolicy.toString())
                                .withPolicyName("crrRolePolicy");

                iamClient.putRolePolicy(putRolePolicyRequest);

        }
}
```

C#

Das folgende AWS SDK for .NET Codebeispiel fügt einem Bucket eine Replikationskonfiguration hinzu und ruft sie dann ab. Um diesen Code zu verwenden, geben Sie die Namen für die Buckets

und den Amazon-Ressourcennamen (ARN) für die IAM-Rolle an. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CrossRegionReplicationTest
    {
        private const string sourceBucket = "*** source bucket ***";
        // Bucket ARN example - arn:aws:s3:::destinationbucket
        private const string destinationBucketArn = "*** destination bucket ARN
 ***";
        private const string roleArn = "*** IAM Role ARN ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint sourceBucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(sourceBucketRegion);
            EnableReplicationAsync().Wait();
        }
        static async Task EnableReplicationAsync()
        {
            try
            {
                ReplicationConfiguration replConfig = new ReplicationConfiguration
                {
                    Role = roleArn,
                    Rules =
                        {
                            new ReplicationRule
                            {
                                Prefix = "Tax",
                                Status = ReplicationRuleStatus.Enabled,
                                Destination = new ReplicationDestination
                                {
```

```
                                    BucketArn = destinationBucketArn
                                }
                            }
                        }
                };

                PutBucketReplicationRequest putRequest = new
PutBucketReplicationRequest
                {
                    BucketName = sourceBucket,
                    Configuration = replConfig
                };

                PutBucketReplicationResponse putResponse = await
s3Client.PutBucketReplicationAsync(putRequest);

                // Verify configuration by retrieving it.
                await RetrieveReplicationConfigurationAsync(s3Client);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }
        private static async Task RetrieveReplicationConfigurationAsync(IAmazonS3
client)
        {
            // Retrieve the configuration.
            GetBucketReplicationRequest getRequest = new GetBucketReplicationRequest
            {
                BucketName = sourceBucket
            };
            GetBucketReplicationResponse getResponse = await
client.GetBucketReplicationAsync(getRequest);
            // Print.
            Console.WriteLine("Printing replication configuration information...");
            Console.WriteLine("Role ARN: {0}", getResponse.Configuration.Role);
            foreach (var rule in getResponse.Configuration.Rules)
```

```
            {
                Console.WriteLine("ID: {0}", rule.Id);
                Console.WriteLine("Prefix: {0}", rule.Prefix);
                Console.WriteLine("Status: {0}", rule.Status);
            }
        }
    }
}
```

Konfigurieren der Replikation, wenn sich Quell- und Ziel-Buckets im Eigentum verschiedener Konten befinden

Das Einrichten der Replikation, wenn sich *Quell*- und *Ziel*-Bucket im Besitz verschiedener befinden, AWS-Konten ähnelt dem Einrichten der Replikation, wenn beide Buckets demselben Konto gehören. Der einzige Unterschied besteht darin, dass der Eigentümer des *Ziel*-Buckets dem Eigentümer des *Quell*-Buckets die Berechtigung zum Replizieren von Objekten gewährt, indem er eine Bucket-Richtlinie hinzufügt.

Weitere Informationen zur Konfiguration der Replikation mit serverseitiger Verschlüsselung mit AWS Key Management Service in kontoübergreifenden Szenarien finden Sie unter Erteilen von zusätzlichen Berechtigungen für kontenübergreifende Szenarien.

So konfigurieren Sie die Replikation, wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten

1. In diesem Beispiel erstellen Sie *Quell*- und *Ziel*-Buckets in zwei verschiedenen AWS-Konten. Sie müssen zwei Anmeldeinformationsprofile für die eingerichtet haben AWS CLI (in diesem Beispiel verwenden wir `acctA` und `acctB` für Profilnamen). Weitere Informationen zum Einrichten der Anmeldeinformations-Profile finden Sie unter Named Profiles (Benannte Profile) im AWS Command Line Interface -Benutzerhandbuch.
2. Folgen Sie den step-by-step Anweisungen unter Konfigurieren für Buckets im selben Konto mit den folgenden Änderungen:

   - Verwenden Sie für alle AWS CLI Befehle im Zusammenhang mit Aktivitäten im *Quell*-Bucket (zum Erstellen des *Quell*-Buckets, Aktivieren des Versionings und Erstellen der IAM-Rolle) das `acctA` Profil . Verwenden Sie das Profil `acctB`, um den *Ziel*-Bucket zu erstellen.
   - Stellen Sie sicher, dass die Berechtigungsrichtlinie den *Quell*- und den *Ziel*-Bucket angibt, die Sie für dieses Beispiel erstellt haben.

3. Fügen Sie in der Konsole die folgende Bucket-Richtlinie für den *Ziel*-Bucket hinzu, um dem Eigentümer des *Quell*-Buckets die Berechtigung zum Replizieren von Objekten zu erteilen. Stellen Sie sicher, dass Sie die Richtlinie bearbeiten, indem Sie die AWS-Konto ID des Eigentümers des *Quell*-Buckets und den Namen des *Ziel*-Buckets angeben.

> **Note**
>
> Wenn Sie das folgende Beispiel verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen. Ersetzen Sie *DOC-EXAMPLE-BUCKET* durch den Namen Ihres Ziel-Buckets. Ersetzen Sie *source-bucket-acct-ID:role/service-role/source-acct-IAM-role* durch die Rolle, die Sie für diese Replikationskonfiguration verwenden.
>
> Wenn Sie die IAM-Servicerolle manuell erstellt haben, legen Sie den Rollenpfad als `role/service-role/` fest, wie im folgenden Richtlinienbeispiel dargestellt. Weitere Informationen finden Sie unter IAM ARNs im IAM-Benutzerhandbuch.

```
{
   "Version":"2012-10-17",
   "Id":"",
   "Statement":[
      {
         "Sid":"Set-permissions-for-objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::source-bucket-acct-ID:role/service-role/source-
acct-IAM-role"
         },
         "Action":["s3:ReplicateObject", "s3:ReplicateDelete"],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      },
      {
         "Sid":"Set permissions on bucket",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::source-bucket-acct-ID:role/service-role/source-
acct-IAM-role"
         },
         "Action":["s3:List*", "s3:GetBucketVersioning", "s3:PutBucketVersioning"],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET"
```

```
        }
    ]
}
```

Wählen Sie den Bucket aus und fügen Sie die Bucket-Richtlinie hinzu. Anweisungen finden Sie unter Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole.

Bei der Replikation besitzt der Eigentümer des Quellobjekts standardmäßig das Replikat. Wenn sich Quell- und Ziel-Bucket im Besitz verschiedener befinden AWS-Konten, können Sie optionale Konfigurationseinstellungen hinzufügen, um die Replikateigentümerschaft in das zu ändern AWS-Konto , das die Ziel-Buckets besitzt. Dazu gehört auch die Gewährung der `ObjectOwnerOverrideToBucketOwner`-Berechtigung. Weitere Informationen finden Sie unter Ändern des Replikat-Eigentümers.

Ändern des Replikat-Eigentümers, wenn sich Quell- und Ziel-Buckets im Eigentum unterschiedlicher Konten befinden

Wenn sich der *`Quell`*- und der *`Ziel`*-Bucket in einer Replikationskonfiguration im Besitz verschiedener befinden AWS-Konten, können Sie Amazon S3 anweisen, die Replikateigentümerschaft in den zu ändern AWS-Konto , der den *`Ziel`*-Bucket besitzt. In diesem Beispiel wird erläutert, wie Sie die Amazon S3-Konsole und die verwenden AWS CLI , um die Replikateigentümerschaft zu ändern. Weitere Informationen finden Sie unter Ändern des Replikat-Eigentümers.

> **Note**
>
> Wenn Sie die S3-Replikation verwenden und sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten, kann der Bucket-Eigentümer des Ziel-Buckets ACLs deaktivieren (mit der Einstellung „Bucket-Eigentümer erzwungen“ für Object Ownership), um die Replikat-Eigentümerschaft in den zu ändern AWS-Konto , der den Ziel-Bucket besitzt. Diese Einstellung ahmt das Verhalten der bestehenden Besitzerüberschreibung nach, ohne dass eine `s3:ObjectOwnerOverrideToBucketOwner`-Berechtigung erforderlich ist. Dies bedeutet, dass alle Objekte, die mit der erzwungenen Einstellung des Bucket-Eigentümers in den Ziel-Bucket repliziert werden, dem Eigentümer des Ziel-Buckets gehören. Informationen zu Object Ownership finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Weitere Informationen zum Konfigurieren der Replikation mit serverseitiger Verschlüsselung mit AWS Key Management Service in kontoübergreifenden Szenarien finden Sie unter Erteilen von zusätzlichen Berechtigungen für kontenübergreifende Szenarien.

Verwenden der S3-Konsole

step-by-step Anweisungen finden Sie unter Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos. Dieses Thema enthält Anweisungen zum Festlegen der Replikationskonfiguration, wenn sich Buckets im Besitz desselben und unterschiedlicher befinden AWS-Konten.

Verwenden der AWS CLI

Um die Replikateigentümerschaft mithilfe der zu ändern AWS CLI, erstellen Sie Buckets, aktivieren das Versioning für die Buckets, erstellen eine IAM-Rolle, die Amazon S3 die Berechtigung zum Replizieren von Objekten gibt, und fügen die Replikationskonfiguration zum Quell-Bucket hinzu. In der Replikations-Konfiguration weisen Sie Amazon S3 an, den Replikatseigentümer zu ändern. Sie testen außerdem die Einrichtung.

So ändern Sie die Replikateigentümerschaft, wenn sich Quell- und Ziel-Bucket im Besitz unterschiedlicher AWS-Konten (AWS CLI) befinden

1. In diesem Beispiel erstellen Sie die *`Quell`*- und *`Ziel`*-Buckets in zwei verschiedenen AWS-Konten. Konfigurieren Sie die AWS CLI mit zwei benannten Profilen. Bei diesem Beispiel werden die Profile `acctA` bzw. `acctB`verwendet. Weitere Informationen zum Einrichten der Anmeldeinformations-Profile finden Sie unter Named Profiles (Benannte Profile) im AWS Command Line Interface -Benutzerhandbuch.

   > **Important**
   >
   > Die Profile, die Sie für diese Übung verwenden, müssen über die nötigen Berechtigungen verfügen. Beispielsweise legen Sie in der Replikations-Konfiguration die IAM-Rolle fest, die Amazon S3 annehmen kann. Dies können Sie nur tun, wenn das verwendete Profil über die `iam:PassRole`-Berechtigung verfügt. Wenn Sie zur Erstellung eines benannten Profils die Benutzer-Anmeldeinformationen eines Administrators verwenden, können Sie alle Aufgaben durchführen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen - AWS Service übergeben kann im IAM-Benutzerhandbuch.

Sie müssen sicherstellen, dass diese Profile über die nötigen Berechtigungen verfügen. Beispielsweise enthält die Replikations-Konfiguration eine IAM-Rolle, die Amazon S3 annehmen kann. Das benannte Profil, mit dem Sie eine solche Konfiguration einem Bucket anfügen, kann dies nur tun, wenn es über die `iam:PassRole`-Berechtigung verfügt. Wenn Sie zur Erstellung dieser benannten Profile die Benutzer-Anmeldeinformationen eines Administrators verwenden, haben diese alle Berechtigungen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen - AWS Service übergeben kann im IAM-Benutzerhandbuch.

2. Erstellen Sie den *`Quell`*-Bucket und aktivieren Sie das Versioning für ihn. In diesem Beispiel erstellen wir den *`Quell`*-Bucket in der Region USA Ost (Nord-Virginia) (us-east-1).

```
aws s3api create-bucket \
--bucket source \
--region us-east-1 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket source \
--versioning-configuration Status=Enabled \
--profile acctA
```

3. Erstellen Sie einen *`Ziel`*-Bucket und aktivieren Sie das Versioning für ihn. In diesem Beispiel erstellen wir den *`Ziel`*-Bucket in der Region US West (Oregon) (us-west-2). Verwenden Sie ein anderes AWS-Konto -Profil als das, das Sie für den *`Quell`*-Bucket verwendet haben.

```
aws s3api create-bucket \
--bucket destination \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctB
```

```
aws s3api put-bucket-versioning \
--bucket destination \
--versioning-configuration Status=Enabled \
--profile acctB
```

4. Sie müssen Ihrer *Ziel*-Bucket-Richtlinie Berechtigungen hinzufügen, um eine Änderung der Replikateigentümerschaft zuzulassen.

   a. Speichern Sie die folgende Richtlinie in *destination-bucket-policy*.json.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "destination_bucket_policy_sid",
      "Principal": {
        "AWS": "source-bucket-owner-account-id"
      },
      "Action": [
        "s3:ReplicateObject",
        "s3:ReplicateDelete",
        "s3:ObjectOwnerOverrideToBucketOwner",
        "s3:ReplicateTags",
        "s3:GetObjectVersionTagging"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::destination/*"
      ]
    }
  ]
}
```

   b. Legen Sie die obige Richtlinie im *Ziel*-Bucket ab:

```
aws s3api put-bucket-policy --region $ {destination_region} --
bucket $ {destination} --policy file://destination_bucket_policy.json
```

5. Erstellen Sie eine IAM-Rolle. Sie geben diese Rolle in der Replizierungskonfiguration an, die Sie später zum *Quell-Bucket* hinzufügen. Amazon S3 übernimmt diese Rolle, um Objekte in Ihrem Namen zu replizieren. Sie erstellen eine IAM-Rolle in zwei Schritten:

   - Erstellen Sie eine Rolle.
   - Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

a. Erstellen Sie eine IAM-Rolle.

i. Kopieren Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-trust-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie erteilt Amazon S3 Berechtigungen für die Übernahme der Rolle.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

ii. Führen Sie den folgenden AWS CLI Befehl aus, um eine Rolle zu erstellen.

```
$ aws iam create-role \
--role-name replicationRole \
--assume-role-policy-document file://s3-role-trust-policy.json  \
--profile acctA
```

b. Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

i. Kopieren Sie die folgende Berechtigungsrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-perm-pol-changeowner.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Zugriffsrichtlinie erteilt Berechtigungen für verschiedene Amazon-S3-Bucket- und -Objektaktionen. In den folgenden Schritten erstellen Sie eine IAM-Rolle und fügen diese Richtlinie der Rolle an.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
```

```
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::source/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration"
         ],
         "Resource":[
            "arn:aws:s3:::source"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ObjectOwnerOverrideToBucketOwner",
            "s3:ReplicateTags",
            "s3:GetObjectVersionTagging"
         ],
         "Resource":"arn:aws:s3:::destination/*"
      }
   ]
}
```

ii. Um eine Richtlinie zu erstellen und sie an die Rolle anzufügen, führen Sie den folgenden Befehl aus:

```
$ aws iam put-role-policy \
--role-name replicationRole \
--policy-document file://s3-role-perm-pol-changeowner.json \
--policy-name replicationRolechangeownerPolicy \
--profile acctA
```

6. Fügen Sie Ihrem Quell-Bucket eine Replikations-Konfiguration hinzu.

a. Der AWS CLI erfordert die Angabe der Replikationskonfiguration als JSON. Speichern Sie den folgenden JSON-Code in einer Datei mit dem Namen `replication.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Fügen Sie in der Konfiguration `AccessControlTranslation` hinzu, um eine Änderung in der Replikateigentümerschaft anzugeben.

```
{
   "Role":"IAM-role-ARN",
   "Rules":[
      {
         "Status":"Enabled",
         "Priority":1,
         "DeleteMarkerReplication":{
            "Status":"Disabled"
         },
         "Filter":{
         },
         "Status":"Enabled",
         "Destination":{
            "Bucket":"arn:aws:s3:::destination",
            "Account":"destination-bucket-owner-account-id",
            "AccessControlTranslation":{
               "Owner":"Destination"
            }
         }
      }
   ]
}
```

b. Bearbeiten Sie den JSON-Code, indem Sie Werte für die Konto-ID des Eigentümers des *Ziel*-Buckets und für *IAM-role-ARN* angeben. Speichern Sie die Änderungen.

c. Um die Replikations-Konfiguration zum Quell-Bucket hinzuzufügen, führen Sie den folgenden Befehl aus. Geben Sie den Namen für den *Quell*-Bucket an.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket source \
--profile acctA
```

7. Prüfen Sie die Replikat-Eigentümerschaft in der Amazon-S3-Konsole.

a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

b. Fügen Sie Objekte zum *Quell*-Bucket hinzu. Stellen Sie sicher, dass der *Ziel*-Bucket die Objektreplikate enthält und dass die Eigentümerschaft der Replikate in den geändert wurde AWS-Konto , der den *Ziel*-Bucket besitzt.

Verwenden der AWS SDKs

Ein Code-Beispiel zum Hinzufügen einer Replikationskonfiguration finden Sie unter Verwenden der AWS SDKs. Sie müssen die Replikations-Konfiguration entsprechend ändern. Weitere konzeptuelle Informationen finden Sie unter Ändern des Replikat-Eigentümers.

Replizieren verschlüsselter Objekte

Standardmäßig repliziert Amazon S3 keine Objekte, die mit serverseitiger Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder serverseitiger Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) verschlüsselt wurden. Um mit SSE-KMS oder DSSE-KMS verschlüsselte Objekte zu replizieren, müssen Sie die Bucket-Replikationskonfiguration ändern und weisen damit Amazon S3 an, diese Objekte zu replizieren. In diesem Beispiel wird erläutert, wie Sie die Amazon S3-Konsole und die AWS Command Line Interface (AWS CLI) verwenden, um die Bucket-Replikationskonfiguration zu ändern und die Replikation verschlüsselter Objekte zu ermöglichen.

Weitere Informationen finden Sie unter Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten.

> **Note**
>
> Wenn ein S3-Bucket-Schlüssel für den Quell- oder Ziel-Bucket aktiviert ist, ist der Verschlüsselungskontext der Amazon-Ressourcenname (ARN) des Buckets, nicht der Objekt-ARN. Sie müssen Ihre IAM-Richtlinien aktualisieren, um den Bucket-ARN für den Verschlüsselungskontext verwenden zu können. Weitere Informationen finden Sie unter S3 Bucket-Schlüssel und Replikation.

> **Note**
>
> Sie können Multi-Region AWS KMS keys in Amazon S3 verwenden. Amazon S3 behandelt jedoch derzeit Multi-Regions-Schlüssel wie Einzel-Regions-Schlüssel und verwendet nicht die Multi-Regions-Funktionen des Schlüssels. Weitere Informationen finden Sie unter Using multi-Region keys (Verwenden von Multi-Regions-Zugriffpunkt-Schlüsseln) im AWS Key Management Service -Entwicklerhandbuch.

Verwenden der S3-Konsole

step-by-step Anweisungen finden Sie unter Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos. Dieses Thema enthält Anweisungen zum Festlegen einer Replikationskonfiguration, wenn sich die Buckets im Besitz desselben und unterschiedlicher befinden AWS-Konten.

Verwenden der AWS CLI

Gehen Sie wie folgt vor AWS CLI, um verschlüsselte Objekte mit der zu replizieren:

- Erstellen Sie Quell- und Ziel-Buckets und aktivieren Sie die Versionsverwaltung für diese Buckets.
- Erstellen Sie eine AWS Identity and Access Management (IAM)-Servicerolle, die Amazon S3 die Berechtigung zum Replizieren von Objekten erteilt. Zu den Berechtigungen für die IAM-Rolle gehören die, die zum Replizieren der verschlüsselten Objekte notwendig sind.
- Fügen Sie eine Replikationskonfiguration zum Quell-Bucket hinzu. Die Replikationskonfiguration stellt Informationen zur Replizierung von Objekten bereit, die mit KMS-Schlüsseln verschlüsselt wurden.
- Fügen Sie verschlüsselte Objekte zum Quell-Bucket hinzu.
- Testen Sie die Einrichtung, um sicherzustellen, dass Ihre verschlüsselten Objekte in den Ziel-Bucket repliziert werden.

Die folgenden Verfahren führen Sie durch diesen Prozess.

Replizieren Sie serverseitig verschlüsselte Objekte (AWS CLI) wie folgt:

1. In diesem Beispiel erstellen Sie den *DOC-EXAMPLE-SOURCE-BUCKET*- und *DOC-EXAMPLE-DESTINATION-BUCKET*-Bucket im selben AWS-Konto. Sie richten auch das

Anmeldeinformationsprofil für die AWS CLI ein. Dieses Beispiel verwendet den Profilnamen *acctA*.

Weitere Informationen zum Festlegen von Anmeldeinformationsprofilen finden Sie unter Benannte Profile im AWS Command Line Interface -Benutzerhandbuch. Die Befehle in diesem Beispiel können Sie verwenden, indem Sie die *user input placeholders* durch Ihre eigenen Informationen ersetzen.

2. Verwenden Sie die folgenden Befehle, um den *DOC-EXAMPLE-SOURCE-BUCKET*-Bucket zu erstellen und die Versionsverwaltung für diesen zu aktivieren. Mit den folgenden Beispielbefehlen wird der *DOC-EXAMPLE-SOURCE-BUCKET*-Bucket in der Region USA Ost (Nord-Virginia) (`us-east-1`) erstellt.

```
aws s3api create-bucket \
--bucket DOC-EXAMPLE-SOURCE-BUCKET \
--region us-east-1 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket DOC-EXAMPLE-SOURCE-BUCKET \
--versioning-configuration Status=Enabled \
--profile acctA
```

3. Verwenden Sie die folgenden Befehle, um den *DOC-EXAMPLE-DESTINATION-BUCKET*-Bucket zu erstellen und die Versionsverwaltung für diesen zu aktivieren. Mit den folgenden Beispielbefehlen wird der *DOC-EXAMPLE-DESTINATION-BUCKET*-Bucket in der Region USA West (Oregon) (`us-west-2`) erstellt.

> **Note**
>
> Um eine Replikationskonfiguration einzurichten, wenn sich sowohl *DOC-EXAMPLE-SOURCE-BUCKET* - als auch -*DOC-EXAMPLE-DESTINATION-BUCKET*Buckets im selben befinden AWS-Konto, verwenden Sie dasselbe Profil. In diesem Beispiel verwenden wir *acctA*. Zum Konfigurieren der Replikation, wenn sich die Buckets im Besitz unterschiedlicher AWS-Konten befinden, legen Sie verschiedene Profile für die Buckets fest.

```
aws s3api create-bucket \
--bucket DOC-EXAMPLE-DESTINATION-BUCKET \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket DOC-EXAMPLE-DESTINATION-BUCKET \
--versioning-configuration Status=Enabled \
--profile acctA
```

4. Als Nächstes erstellen Sie eine IAM-Servicerolle. Sie werden diese Rolle in der Replikationskonfiguration angeben, die Sie später zum Bucket *DOC-EXAMPLE-SOURCE-BUCKET* hinzufügen. Amazon S3 übernimmt diese Rolle, um Objekte in Ihrem Namen zu replizieren. Sie erstellen eine IAM-Rolle in zwei Schritten:

   - Erstellen Sie eine Servicerolle.
   - Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

   a. Um eine IAM-Servicerolle zu erstellen, gehen Sie wie folgt vor:

      i. Kopieren Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-trust-policy-kmsobj.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie weist dem Amazon-S3-Service-Prinzipal die Berechtigungen zum Annehmen der Rolle zu, damit Amazon S3 Aufgaben in Ihrem Namen durchführen kann.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
```

```
}
```

ii. Verwenden Sie den folgenden Befehl, um die Rolle zu erstellen:

```
$ aws iam create-role \
--role-name replicationRolekmsobj \
--assume-role-policy-document file://s3-role-trust-policy-kmsobj.json  \
--profile acctA
```

b. Als Nächstes fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu. Diese Zugriffsrichtlinie erteilt Berechtigungen für verschiedene Amazon-S3-Bucket- und -Objektaktionen.

i. Kopieren Sie die folgende Berechtigungsrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-permissions-policykmsobj.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Sie werden eine IAM-Rolle erstellen und fügen ihr später die Richtlinie an.

> **Important**
>
> In der Berechtigungsrichtlinie geben Sie die AWS KMS Schlüssel-IDs an, die für die Verschlüsselung der *DOC-EXAMPLE-DESTINATION-BUCKET* Buckets *DOC-EXAMPLE-SOURCE-BUCKET* und verwendet werden. Sie müssen zwei separate KMS-Schlüssel für die Buckets *DOC-EXAMPLE-SOURCE-BUCKET* und *DOC-EXAMPLE-DESTINATION-BUCKET* erstellen. AWS KMS keys werden nicht außerhalb der geteilt, AWS-Region in der sie erstellt wurden.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration",
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl",
            "s3:GetObjectVersionTagging"
         ],
         "Effect":"Allow",
         "Resource":[
```

```
            "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
            "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET/*"
         ]
      },
      {
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLikeIfExists":{
               "s3:x-amz-server-side-encryption":[
                  "aws:kms",
                  "AES256",
                  "aws:kms:dsse"
               ],
               "s3:x-amz-server-side-encryption-aws-kms-key-id":[
                  "AWS KMS key IDs(in ARN format) to use for encrypting
object replicas"
               ]
            }
         },
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/*"
      },
      {
         "Action":[
            "kms:Decrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.us-east-1.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET/*"
               ]
            }
         },
         "Resource":[
            "AWS KMS key IDs(in ARN format) used to encrypt source
objects."
         ]
      },
```

```
      {
         "Action":[
            "kms:Encrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.us-west-2.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/*"
               ]
            }
         },
         "Resource":[
            "AWS KMS key IDs(in ARN format) to use for encrypting object
 replicas"
         ]
      }
   ]
}
```

ii. Erstellen Sie eine Richtlinie und hängen Sie sie an die Rolle an.

```
$ aws iam put-role-policy \
--role-name replicationRolekmsobj \
--policy-document file://s3-role-permissions-policykmsobj.json \
--policy-name replicationRolechangeownerPolicy \
--profile acctA
```

5. Als Nächstes fügen Sie die folgende Replikationskonfiguration zum *DOC-EXAMPLE-SOURCE-BUCKET*-Bucket hinzu. Sie weist Amazon S3 an, Objekte mit dem Präfix `Tax/` in den *DOC-EXAMPLE-DESTINATION-BUCKET*-Bucket zu replizieren.

> **Important**
>
> In der Replikationskonfiguration legen Sie die IAM-Rolle fest, die Amazon S3 annehmen kann. Dies können Sie nur tun, wenn Sie über die `iam:PassRole`-Berechtigung verfügen. Das Profil, das Sie im CLI-Befehl angeben, muss über diese Berechtigung verfügen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Serviceübergeben kann im IAM-Benutzerhandbuch.

```
 <ReplicationConfiguration>
  <Role>IAM-Role-ARN</Role>
  <Rule>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>Disabled</Status>
    </DeleteMarkerReplication>
    <Filter>
       <Prefix>Tax</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <SourceSelectionCriteria>
      <SseKmsEncryptedObjects>
        <Status>Enabled</Status>
      </SseKmsEncryptedObjects>
    </SourceSelectionCriteria>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET</Bucket>
      <EncryptionConfiguration>
        <ReplicaKmsKeyID>AWS KMS key IDs to use for encrypting object replicas</
ReplicaKmsKeyID>
      </EncryptionConfiguration>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Um eine Replikationskonfiguration zum *DOC-EXAMPLE-SOURCE-BUCKET*-Bucket hinzuzufügen, gehen Sie wie folgt vor:

a. Der AWS CLI erfordert, dass Sie die Replikationskonfiguration als JSON angeben. Speichern Sie den folgenden JSON-Code in einer Datei (`replication.json`) im aktuellen Verzeichnis auf Ihrem lokalen Computer.

```
{
   "Role":"IAM-Role-ARN",
   "Rules":[
      {
         "Status":"Enabled",
         "Priority":1,
         "DeleteMarkerReplication":{
```

```
            "Status":"Disabled"
         },
         "Filter":{
            "Prefix":"Tax"
         },
         "Destination":{
            "Bucket":"arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET",
            "EncryptionConfiguration":{
               "ReplicaKmsKeyID":"AWS KMS key IDs (in ARN format) to use for
 encrypting object replicas"
            }
         },
         "SourceSelectionCriteria":{
            "SseKmsEncryptedObjects":{
               "Status":"Enabled"
            }
         }
      }
   ]
}
```

b. Bearbeiten Sie den JSON-Code, um Werte für den *DOC-EXAMPLE-DESTINATION-BUCKET*-Bucket, die *AWS KMS key IDs (in ARN format)* und den *IAM-role-ARN* anzugeben. Speichern Sie die Änderungen.

c. Verwenden Sie den folgenden Befehl, um die Replikationskonfiguration zu Ihrem *DOC-EXAMPLE-SOURCE-BUCKET*-Bucket hinzuzufügen. Stellen Sie sicher, dass Sie den Namen für den *DOC-EXAMPLE-SOURCE-BUCKET*-Bucket angeben.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket DOC-EXAMPLE-SOURCE-BUCKET \
--profile acctA
```

6. Testen Sie die Konfiguration, um zu überprüfen, ob die verschlüsselten Objekte repliziert werden. Führen Sie in der Amazon-S3-Konsole die folgenden Schritte aus:

a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

b. Erstellen Sie im *DOC-EXAMPLE-SOURCE-BUCKET*-Bucket einen Ordner mit dem Namen `Tax`.

c. Fügen Sie Beispielobjekte zum Ordner hinzu. Stellen Sie sicher, dass Sie die Verschlüsselungsoption wählen und Ihren KMS-Schlüssel angeben, um die Objekte zu verschlüsseln.

d. Vergewissern Sie sich, dass der *DOC-EXAMPLE-DESTINATION-BUCKET*-Bucket die Objektreplikate enthält und dass sie mithilfe des KMS-Schlüssels verschlüsselt sind, den Sie in der Konfiguration angegeben haben. Weitere Informationen finden Sie unter the section called “Abrufen des Replikationsstatus”.

Verwenden der AWS SDKs

Ein Code-Beispiel, das zeigt, wie eine Replikationskonfiguration hinzugefügt wird, finden Sie unter Verwenden der AWS SDKs. Sie müssen die Replikationskonfiguration entsprechend ändern.

Weitere konzeptuelle Informationen finden Sie unter Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten.

Replizieren von Objekten mit S3-Replikationszeitkontrolle (S3 RTC)

S3 Replication Time Control (S3 RTC) hilft Ihnen bei der Einhaltung von Compliance- oder Geschäftsanforderungen für die Datenreplikation und bietet Einblick in die Amazon-S3-Replikationsaktivitäten. S3 RTC repliziert die meisten Objekte, die Sie zu Amazon S3 hochladen, in Sekunden und 99,99 Prozent dieser Objekte innerhalb von 15 Minuten.

Mit S3 RTC können Sie die Gesamtzahl und Größe der Objekte, die zur Replikation ausstehen, sowie die maximale Replikationszeit in die Zielregion überwachen. Replikationsmetriken sind über die AWS Management Console und das Amazon- CloudWatch Benutzerhandbuch verfügbar. Weitere Informationen finden Sie unter the section called “S3-Replikationsmetriken in CloudWatch ”.

Verwenden der S3-Konsole

step-by-step Anweisungen finden Sie unter Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos. Dieses Thema enthält Anweisungen zum Aktivieren von S3 RTC in Ihrer Replikationskonfiguration, wenn sich Buckets im Besitz desselben und unterschiedlicher befinden AWS-Konten.

Verwenden der AWS CLI

Um Objekte mit aktiviertem S3 RTC AWS CLI zu replizieren, erstellen Sie Buckets, aktivieren das Versioning für die Buckets, erstellen eine IAM-Rolle, die Amazon S3 die Berechtigung zum

Replizieren von Objekten erteilt, und fügen die Replikationskonfiguration zum Quell-Bucket hinzu. In der Replikations-Konfiguration muss S3-Replikationszeitkontrolle (S3 RTC) aktiviert sein.

Replizieren Sie mit aktiviertem S3 RTC (AWS CLI) wie folgt:

- Im folgenden Beispiel werden `ReplicationTime` und `Metric` festgelegt und die Replikationskonfiguration zum Quell-Bucket hinzugefügt.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "DeleteMarkerReplication": {
                "Status": "Disabled"
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::destination",
                "Metrics": {
                    "Status": "Enabled",
                    "EventThreshold": {
                        "Minutes": 15
                    }
                },
                "ReplicationTime": {
                    "Status": "Enabled",
                    "Time": {
                        "Minutes": 15
                    }
                }
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

> **Important**
>
> Für `Metrics:EventThreshold:Minutes` und `ReplicationTime:Time:Minutes` ist als gültiger Wert nur 15 zulässig.

Verwenden des AWS SDK for Java

Nachfolgend finden Sie ein Java-Beispiel zum Hinzufügen einer Replikations-Konfiguration mit der S3-Replikationszeitkontrolle (S3 RTC).

```
import software.amazon.awssdk.auth.credentials.AwsBasicCredentials;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.model.DeleteMarkerReplication;
import software.amazon.awssdk.services.s3.model.Destination;
import software.amazon.awssdk.services.s3.model.Metrics;
import software.amazon.awssdk.services.s3.model.MetricsStatus;
import software.amazon.awssdk.services.s3.model.PutBucketReplicationRequest;
import software.amazon.awssdk.services.s3.model.ReplicationConfiguration;
import software.amazon.awssdk.services.s3.model.ReplicationRule;
import software.amazon.awssdk.services.s3.model.ReplicationRuleFilter;
import software.amazon.awssdk.services.s3.model.ReplicationTime;
import software.amazon.awssdk.services.s3.model.ReplicationTimeStatus;
import software.amazon.awssdk.services.s3.model.ReplicationTimeValue;

public class Main {

  public static void main(String[] args) {
    S3Client s3 = S3Client.builder()
      .region(Region.US_EAST_1)
      .credentialsProvider(() -> AwsBasicCredentials.create(
          "AWS_ACCESS_KEY_ID",
          "AWS_SECRET_ACCESS_KEY")
      )
      .build();

    ReplicationConfiguration replicationConfig = ReplicationConfiguration
      .builder()
      .rules(
          ReplicationRule
            .builder()
            .status("Enabled")
```

```
            .priority(1)
            .deleteMarkerReplication(
                DeleteMarkerReplication
                    .builder()
                    .status("Disabled")
                    .build()
            )
            .destination(
                Destination
                    .builder()
                    .bucket("destination_bucket_arn")
                    .replicationTime(
                        ReplicationTime.builder().time(
                            ReplicationTimeValue.builder().minutes(15).build()
                        ).status(
                            ReplicationTimeStatus.ENABLED
                        ).build()
                    )
                    .metrics(
                        Metrics.builder().eventThreshold(
                            ReplicationTimeValue.builder().minutes(15).build()
                        ).status(
                            MetricsStatus.ENABLED
                        ).build()
                    )
                    .build()
            )
            .filter(
                ReplicationRuleFilter
                    .builder()
                    .prefix("testtest")
                    .build()
            )
        .build())
        .role("role_arn")
        .build();

    // Put replication configuration
    PutBucketReplicationRequest putBucketReplicationRequest =
PutBucketReplicationRequest
      .builder()
      .bucket("source_bucket")
      .replicationConfiguration(replicationConfig)
      .build();
```

```
    s3.putBucketReplication(putBucketReplicationRequest);
  }
}
```

Weitere Informationen finden Sie unter Erfüllen der Compliance-Anforderungen mit S3-Replikationszeitkontrolle (S3 RTC).

Verwalten von Replikationsregeln mit der Amazon-S3-Konsole

Die Replikation ist das automatische, asynchrone Kopieren von Objekten über Buckets hinweg in derselben oder einer anderen AWS-Regionen. Sie repliziert neu erstellte Objekte und Objektaktualisierungen aus einem Quell-Bucket in einen angegebenen Ziel-Bucket.

Sie verwenden die Amazon-S3-Konsole, um dem Quell-Bucket Replikationsregeln hinzuzufügen. Replikationsregeln definieren, welche Quell-Bucket-Objekte repliziert werden sollen, und den Ziel-Bucket/die Ziel-Buckets, in dem/denen die replizierten Objekte gespeichert werden sollen. Weitere Informationen zur Replikation finden Sie unter Replizieren von Objekten.

Sie können Replikationsregeln auf der Seite Replication (Replikation) verwalten. Sie können Replikationsregeln hinzufügen, anzeigen, aktivieren, deaktivieren und löschen sowie ihre Priorität ändern. Weitere Informationen zum Hinzufügen von Replikationsregeln zu einem Bucket finden Sie unter Verwenden der S3-Konsole.

So werden die Replikationsregeln für einen S3-Bucket verwaltet

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des von Ihnen erstellten Buckets aus.
3. Wählen Sie Management (Verwaltung) und scrollen Sie dann nach unten zu Replication rules (Replikationsregeln).
4. Sie können die Replikationsregeln wie folgt ändern.
   - Um eine Replikationsregel zu aktivieren oder zu deaktivieren, wählen Sie die Regel und dann Actions (Aktionen) aus. Anschließend wählen Sie in der Dropdown-Liste Enable rule (Regel aktivieren) oder Disable rule (Regel deaktivieren) aus. Sie können in der Dropdown-Liste Actions (Aktionen) auch alle Regeln in dem Bucket deaktivieren, aktivieren oder löschen.

- Um eine Replikationsregel zu ändern, wählen Sie die Regel und dann Edit (Bearbeiten) aus. Hierdurch wird der Replikationsassistent gestartet, der Sie durch den Vorgang führt. Weitere Informationen zur Verwendung des Assistenten finden Sie unter Verwenden der S3-Konsole.

  Regelprioritäten werden festgelegt, um Konflikte zu vermeiden, die durch Objekte verursacht werden, die mehreren Regeln unterliegen. Wenn sich Regeln überschneiden, verwendet Amazon S3 die Regelpriorität, um die Regel zu ermitteln, die angewendet werden muss. Je höher die Zahl, desto höher die Priorität. Weitere Informationen zur Priorität von Regeln finden Sie unter Replikations-Konfiguration.

Replizieren bestehender Objekte mit S3-Batch-Replikation

Die S3-Batch-Replikation bietet Ihnen eine Möglichkeit, Objekte zu replizieren, die existierten, bevor eine Replikationskonfiguration vorhanden war, Objekte, die zuvor repliziert wurden, und Objekte, bei denen die Replikation fehlgeschlagen ist. Dies geschieht durch die Verwendung eines Batchvorgang-Auftrags. Dies unterscheidet sich von der Live-Replikation, die kontinuierlich und automatisch neue Objekte in Amazon-S3-Buckets hinweg repliziert. Um mit der Batch-Replikation zu beginnen, können Sie folgende Aktionen ausführen:

- Initiieren der Batch-Replikation für eine neue Replikationsregel oder ein neues Ziel – Sie können einen einmaligen Batch-Replikationsauftrag erstellen, wenn Sie die erste Regel in einer neuen Replikationskonfiguration erstellen oder wenn Sie ein neues Ziel zu einer vorhandenen Konfiguration über die AWS Management Console erstellen.
- Initiieren der Batch-Replikation für eine vorhandene Replikationskonfiguration – Sie können einen neuen Batch-Replikationsauftrag mithilfe von S3-Batchoperationen über die - AWS SDKs , AWS Command Line Interface (AWS CLI) oder die Amazon S3-Konsole erstellen.

Wenn der Batch-Replikationsauftrag abgeschlossen ist, erhalten Sie einen Abschlussbericht. Weitere Informationen dazu, wie Sie den Bericht verwenden können, um den Auftrag zu untersuchen, finden Sie unter Verfolgen von Auftragsstatus- und Abschluss.

Überlegungen zur S3-Batch-Replikation

- Ihr Quell-Bucket muss über eine vorhandene Replikationskonfiguration verfügen. Informationen zum Aktivieren der Replikation finden Sie unter Einrichten der Replikation und Anleitungen: Beispiele zum Konfigurieren der Replikation.

- Wenn Sie S3 Lifecycle für Ihren Bucket konfiguriert haben, empfehlen wir, Ihre Lifecycle-Regeln zu deaktivieren, während der Batch-Replikationsauftrag aktiv ist. Dadurch wird die Parität zwischen Quell- und Ziel-Bucket sichergestellt. Andernfalls könnten diese Buckets voneinander abweichen und der Ziel-Bucket wäre keine exakte Kopie des Quell-Buckets. Berücksichtigen Sie dabei Folgendes:
  - Ihr Quell-Bucket hat mehrere Versionen für ein Objekt und eine Löschmarkierung.
  - Ihre Quell- und Ziel-Buckets verfügen über eine Lebenszyklus-Konfiguration zum Entfernen abgelaufener Löschmarkierungen.

  Die Batch-Replikation repliziert möglicherweise die Löschmarkierung in den Ziel-Bucket, bevor die Objektversionen repliziert werden. Dies kann dazu führen, dass die Löschmarkierung als abgelaufen markiert und aus dem Ziel-Bucket entfernt wird, bevor die Objekte kopiert werden.
- Die AWS Identity and Access Management (IAM)-Rolle, die Sie zum Ausführen des Batchoperationenauftrags angeben, muss über Berechtigungen zum Ausführen des zugrunde liegenden Batchreplikationsvorgangs verfügen. Weitere Informationen zum Erstellen einer IAM-Rolle finden Sie unter Konfigurieren von IAM-Richtlinien für die Batch-Replikation.
- Die Batch-Replikation erfordert ein Manifest, das von Amazon S3 generiert werden kann. Das generierte Manifest muss in derselben AWS-Region wie der Quell-Bucket gespeichert werden. Wenn Sie das Manifest nicht generieren möchten, können Sie einen Amazon-S3-Bestandsbericht oder eine CSV-Datei bereitstellen, die die Objekte enthält, die Sie replizieren möchten.
- Die Batch-Replikation unterstützt nicht das erneute Replizieren von Objekten, die mit der Versions-ID des Objekts aus dem Ziel-Bucket gelöscht wurden. Wenn Sie diese Objekte erneut replizieren möchten, können Sie die Quellobjekte mit einem Batch–Kopierauftrag kopieren. Wenn Sie diese Objekte kopieren, werden neue Versionen des Objekts im Quell-Bucket erstellt und die Replikation zum Ziel wird automatisch initiiert. Durch das Löschen und Neuerstellen des Ziel-Buckets wird keine Replikation initiiert.

  Weitere Informationen zur Batch-Kopie finden Sie unter Beispiele, die Batch-Vorgänge zum Kopieren von Objekten verwenden.
- Wenn Sie eine Replikationsregel für den S3-Bucket verwenden, stellen Sie sicher, dass Sie Ihre Replikationskonfiguration aktualisieren und der IAM-Rolle, die der Replikationsregel zugeordnet ist, die entsprechenden Berechtigungen zum Replizieren von Objekten gewähren. Die IAM-Rolle muss über Berechtigungen zum Ausführen der S3-Aktion sowohl für den Quell- als auch für den Ziel-Bucket verfügen.
- Wenn Sie innerhalb eines kurzen Zeitraums mehrere Batch-Replikationsaufträge für denselben Bucket einreichen, führt S3 diese Aufträge gleichzeitig aus.

- Wenn Sie mehrere Batch-Replikationsaufträge für zwei verschiedene Buckets einreichen, kann es sein, dass S3 nicht alle Aufträge gleichzeitig ausführt. Wenn Sie die Anzahl der Batch-Replikationsaufträge, die in Ihrem Konto gleichzeitig ausgeführt werden können, überschreiten, unterbricht S3 die Aufträge mit niedrigerer Priorität, um die Aufträge mit höherer Priorität zu bearbeiten. Nach Abschluss der Elemente mit höherer Priorität werden alle angehaltenen Aufträge wieder aktiv.
- Die Batchreplikation wird nicht für Objekte unterstützt, die in den S3 Glacier-Speicherklassen „Flexible Retrieval“ und „Deep Archive“ gespeichert wurden.
- Zur Massenreplizierung von S3 Intelligent-Tiering-Objekten, die in der Speicherebene „Archive Access“ oder „Deep Archive Access“ gespeichert sind, müssen Sie eine Anforderung zur Wiederherstellung initiieren und warten, bis die Objekte in die Ebene „Frequent Access“ verschoben wurden.

Angeben eines Manifests für einen Batch-Replikationsauftrag

Ein Manifest ist ein Amazon-S3-Objekt, das Objektschlüssel enthält, die Amazon S3 bearbeiten soll. Wenn Sie einen Batch-Replikationsauftrag erstellen möchten, müssen Sie entweder ein benutzergeneriertes Manifest angeben oder Amazon S3 ein Manifest basierend auf Ihrer Replikationskonfiguration generieren lassen.

Wenn Sie ein benutzergeneriertes Manifest angeben, muss es in Form eines Amazon-S3-Bestandsberichts oder einer CSV-Datei vorliegen. Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, müssen Sie die Versions-IDs für die Objekte angeben. Es wird nur das Objekt mit der im Manifest angegebenen Versions-ID repliziert. Weitere Informationen zum Angeben eines Manifests finden Sie unter Angeben eines Manifests.

Wenn Amazon S3 eine Manifestdatei in Ihrem Namen generieren soll, verwenden die aufgeführten Objekte denselben Quell-Bucket, das gleiche Präfix und die gleichen Tags wie all Ihre Replikationskonfigurationen des Quell-Buckets. Mit einem generierten Manifest repliziert Amazon S3 alle berechtigten Versionen Ihrer Objekte.

> **Note**
>
> Wenn Sie das Manifest generieren möchten, muss es in derselben AWS-Region wie der Quell-Bucket gespeichert werden.

Filter für einen Batch-Replikationsauftrag

Wenn Sie Ihren Batch-Replikationsauftrag erstellen, können Sie optional zusätzliche Filter angeben, z. B. das Erstellungsdatum des Objekts und den Replikationsstatus, um den Umfang des Auftrags zu reduzieren.

Sie können Objekte filtern, die repliziert werden sollen, basierend auf dem `ObjectReplicationStatuses`-Wert, indem Sie einen oder mehrere der folgenden Werte angeben:

- `"NONE"` – Zeigt an, dass Amazon S3 noch nie versucht hat, das Objekt zu replizieren.
- `"FAILED"` – Zeigt an, dass Amazon S3 versucht hat das Objekt zu replizieren, es jedoch zuvor nicht replizieren konnte.
- `"COMPLETED"` – Zeigt an, dass Amazon S3 das Objekt zuvor erfolgreich repliziert hat.
- `"REPLICA"` – Zeigt an, dass dies ein Replikatobjekt ist, das Amazon S3 von einer anderen Quelle repliziert hat.

Weitere Informationen zum Replikationsstatus finden Sie unter Abrufen von Replikationsstatusinformationen.

Wenn Sie nicht nach dem Replikationsstatus filtern, wird der Batchvorgang versuchen, alle berechtigten Objekte zu replizieren. Abhängig von Ihrem Ziel können Sie `ObjectReplicationStatuses` auf einen der folgenden Werte setzen:

- Wenn Sie nur vorhandene Objekte replizieren möchten, die nie repliziert wurden, behalten Sie nur `"NONE"` bei.
- Wenn Sie versuchen möchten, nur Objekte zu replizieren, die zuvor nicht repliziert wurden, behalten Sie nur `"FAILED"` bei.
- Wenn Sie sowohl vorhandene Objekte replizieren als auch versuchen möchten, Objekte zu replizieren, die zuvor nicht repliziert wurden, behalten Sie `"NONE"` und `"FAILED"` bei.
- Wenn Sie einen Ziel-Bucket mit Objekten füllen möchten, die an ein anderes Ziel repliziert wurden, behalten Sie `"COMPLETED"` bei.
- Wenn Sie zuvor replizierte Objekte replizieren möchten, behalten Sie `"REPLICA"` bei.

Abschlussbericht zur Batchreplikation

Wenn Sie einen Batchreplikationsauftrag erstellen, können Sie einen CSV-Abschlussbericht anfordern. In diesem Bericht werden Objekte, Erfolgs- oder Fehlercodes, Ausgaben und Beschreibungen der Replikation angezeigt. Weitere Informationen zum Nachverfolgen von Aufträgen und zu Abschlussberichten finden Sie unter Abschlussberichte.

Eine Liste der Fehlercodes und Beschreibungen der Replikation finden Sie unter Gründe für das Fehlschlagen der Replikation in Amazon S3.

Erste Schritte mit der Batch-Replikation

Weitere Informationen zur Verwendung der Batch-Replikation finden Sie unter Tutorial: Replizieren vorhandener Objekte in Ihren-Amazon S3-Buckets mit S3 Batch Replication.

Konfigurieren von IAM-Richtlinien für die Batch-Replikation

Da die S3-Batch-Replikation eine Art Batchvorgangs-Auftrag ist, müssen Sie eine AWS Identity and Access Management (IAM)-Rolle für Batchvorgänge erstellen, um Amazon-S3-Berechtigungen zum Ausführen von Aktionen in Ihrem Namen zu erteilen. Sie müssen auch eine IAM-Richtlinie für die Batch-Replikation an die IAM-Rolle von Batchvorgängen anhängen. Im folgenden Beispiel wird eine IAM-Rolle erstellt, die Batchvorgängen die Berechtigung zum Initiieren eines Batch-Replikationsauftrags erteilt.

Erstellen Sie eine IAM-Rolle und -Richtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie unter Access management (Zugriffsverwaltung) Roles (Rollen) aus.
3. Wählen Sie Create Role (Rolle erstellen) aus.
4. Wählen Sie AWS-Service als Typ der vertrauenswürdigen Entität, Amazon S3 als Service und S3 Batch Operations (S3-Batchvorgänge) als den Anwendungsfall.
5. Wählen Sie Next: Permissions aus.
6. Wählen Sie Create Policy (Richtlinie erstellen) aus.
7. Wählen Sie JSON und fügen Sie eine der folgenden Richtlinien basierend auf Ihrem Manifest ein.

> **Note**
>
> Wenn Sie ein Manifest erstellen oder ein Manifest liefern, sind andere Berechtigungen erforderlich. Weitere Informationen finden Sie unter Angeben eines Manifests für einen Batch-Replikationsauftrag.

Richtlinie bei Verwendung und Speicherung eines von S3 generierten Manifests

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
      {
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:PutInventoryConfiguration"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***"
         ]
      },
      {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
```

```
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject"
         ],
         "Resource":[
            "arn:aws:s3:::*** completion report bucket ****/*",
            "arn:aws:s3:::*** manifest bucket ****/*"
         ]
      }
   ]
}
```

Richtlinie bei Verwendung eines vom Benutzer bereitgestellten Manifests

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
            {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject"
         ],
         "Resource":[
```

```
            "arn:aws:s3:::*** completion report bucket ****/*"
         ]
      }
   ]
}
```

8. Wählen Sie Next: Tags (Weiter: Tags) aus.
9. Klicken Sie auf Weiter: Prüfen.
10. Wählen Sie einen Namen für die Richtlinie und wählen Sie dann Create policy (Richtlinie erstellen) aus.
11. Fügen Sie diese Richtlinie an Ihre Rolle an und wählen Sie Next: Tags (Weiter: Tags).
12. Wählen Sie Weiter: Prüfen aus.
13. Wählen Sie einen Namen für die Rolle und wählen Sie Create role (Rolle erstellen).

Überprüfen der Vertrauensrichtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie unter Access management (Zugriffsverwaltung) Roles (Rollen) und Ihre neu erstellte Rolle aus.
3. Wählen Sie auf der Registerkarte Trust relationships (Vertrauensstellungen) die Option Edit trust relationship (Vertrauensstellung bearbeiten).
4. Stellen Sie sicher, dass diese Rolle die folgende Vertrauensrichtlinie verwendet:

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

Erstellen eines Batch-Replikationsauftrags für eine erste Replikationsregel oder ein neues Ziel

Wenn Sie die erste Regel in einer neuen Replikationskonfiguration erstellen oder einer vorhandenen Konfiguration über die ein neues Ziel hinzufügen AWS Management Console, können Sie optional einen Batch-Replikationsauftrag erstellen.

Informationen zum Verwenden der Batch-Replikation für eine vorhandene Konfiguration ohne Hinzufügen eines neuen Ziels finden Sie unter Erstellen eines Batch-Replikationsauftrags für vorhandene Replikationsregeln.

Verwenden der Batch-Replikation für eine neue Replikationsregel oder ein neues Ziel über die AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste von Buckets den Namen des Buckets aus, der die Objekte enthält, die Sie replizieren möchten.
3. Wählen Sie zum Erstellen einer neuen Replikationsregel oder zum Bearbeiten einer vorhandenen Regel Management (Verwaltung) und scrollen Sie nach unten zu Replication rules (Replikationsregeln):

   - Wählen Sie zum Erstellen einer neuen Replikationsregel Create replication rule (Erstellen einer Replikationsregel).

     > **Note**
     >
     > Beispiele zum Einrichten einer grundlegenden Replikationsregel finden Sie unter: Anleitungen: Beispiele zum Konfigurieren der Replikation.

   - Wählen Sie zum Bearbeiten einer vorhandenen Replikationsregel die entsprechende Regel aus und wählen Sie dann Edit rule (Regel bearbeiten).
4. Erstellen Sie Ihre neue Replikationsregel oder bearbeiten Sie das Ziel für Ihre vorhandene Replikationsregel und wählen Sie Save (Speichern).

   Nachdem Sie die erste Regel in einer neuen Replikationskonfiguration erstellt oder eine vorhandene Konfiguration bearbeitet haben, um ein neues Ziel hinzuzufügen, wird der Dialog

Replicate existing objects? (Vorhandene Objekte replizieren?) angezeigt, über den Sie einen Batch-Replikationsauftrag erstellen können.

5. Wenn Sie diesen Auftrag jetzt ausführen möchten, wählen Sie Ja, vorhandene Objekte replizieren aus.

   Wenn Sie diesen Auftrag zu einem späteren Zeitpunkt ausführen möchten, wählen Sie Nein, keine vorhandenen Objekte replizieren aus.

6. Erstellen Sie Ihren S3-Batch-Replikationsauftrag. Der S3-Batch-Replikationsauftrag hat mehrere Einstellungen:

   Option zur Auftragsausführung

   Wenn der S3-Batch-Replikationsauftrag sofort ausgeführt werden soll, können Sie Job runs automatically when ready (Auftrag wird automatisch ausgeführt, wenn er bereit ist) wählen. Wenn Sie den Auftrag zu einem späteren Zeitpunkt ausführen möchten, wählen Sie Job waits to be run when ready (Auftrag wartet auf Ausführung, wenn er bereit ist).

   Wenn Sie Job runs automatically when ready (Auftrag wird automatisch ausgeführt, wenn er bereit ist) wählen, können Sie kein Batchvorgangs-Manifest erstellen und speichern. Um das Batchvorgangs-Manifest zu speichern, wählen Sie Job waits to be run when ready (Auftrag wartet auf Ausführung, wenn er bereit ist) aus.

   Batchvorgangs-Manifest

   Das Manifest ist eine Liste aller Objekte, für die die festgelegte Aktion ausgeführt werden soll. Sie können das Batchvorgangs-Manifest speichern. Ähnlich wie bei S3-Inventory-Dateien wird das Manifest als CSV-Datei in einem Bucket gespeichert. Weitere Informationen über die Batchvorgangs-Manifeste finden Sie unter Angeben eines Manifests.

   Abschlussbericht

   S3-Batchvorgänge führen eine Aufgabe für jedes Objekt aus, das im Manifest angegeben ist. Abschlussberichte bieten eine einfache Möglichkeit, die Ergebnisse von Aufgaben in einem konsolidierten Format ohne zusätzliche Einrichtung anzuzeigen. Sie können einen Abschlussbericht für alle oder nur für die fehlgeschlagenen Aufgaben anfordern. Weitere Informationen zu Abschlussberichten finden Sie unter Abschlussberichte.

Berechtigungen

Eine der häufigsten Ursachen für Replikationsfehler sind unzureichende Berechtigungen in der bereitgestellten AWS Identity and Access Management (IAM)-Rolle. Weitere Informationen zum Erstellen dieser Rolle finden Sie unter Konfigurieren von IAM-Richtlinien für die Batch-Replikation.

7. Wählen Sie Create Batch Operations Job (Batchvorgangs-Auftrag erstellen).

Erstellen eines Batch-Replikationsauftrags für vorhandene Replikationsregeln

Sie können die S3-Batch-Replikation für eine vorhandene Replikationskonfiguration konfigurieren, indem Sie die - AWS SDKs , AWS Command Line Interface (AWS CLI) oder die Amazon S3-Konsole verwenden. Eine Übersicht über die Batch-Replikation finden Sie unter Replizieren bestehender Objekte mit S3-Batch-Replikation.

Als Voraussetzung müssen Sie eine IAM-Rolle AWS Identity and Access Management (Batch Operations) erstellen, um Amazon S3 Berechtigungen zum Ausführen von Aktionen in Ihrem Namen zu erteilen, siehe Konfigurieren von IAM-Richtlinien für die Batch-Replikation.

Wenn der Batch-Replikationsauftrag abgeschlossen ist, erhalten Sie einen Abschlussbericht. Weitere Informationen dazu, wie Sie den Bericht verwenden können, um den Auftrag zu untersuchen, finden Sie unter Verfolgen von Auftragsstatus- und Abschluss.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie im Navigationsbereich der Amazon-S3-Konsole auf Batch-Vorgänge (Batch-Vorgänge).
3. Wählen Sie Create job (Auftrag erstellen) aus.
4. Wählen Sie die Region aus, in der Sie Ihren Auftrag erstellen möchten.
5. Wählen Sie das Manifest format (Manifestformat). In diesem Beispiel wird gezeigt, wie ein Manifest basierend auf einer vorhandenen S3-Replikationskonfiguration erstellt wird.

> **Note**
>
> Das Manifest ist eine Liste aller Objekte, für die die festgelegte Aktion ausgeführt werden soll. Weitere Informationen über die Batchvorgangs-Manifeste finden Sie unter Angeben

eines Manifests. Wenn Sie ein Manifest vorbereitet haben, wählen Sie S3 inventory report (manifest.json) (S3 Bestandsbericht) oder CSV aus. Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, sollten Sie die Versions-IDs für die Objekte angeben. Weitere Informationen zum Erstellen eines Manifests finden Sie unter Angeben eines Manifests.

6. Um ein Manifest basierend auf Ihrer Replikationskonfiguration zu erstellen, wählen Sie Create manifest using S3 Replication configuration (Manifest mit der S3-Replikationskonfiguration erstellen). Wählen Sie dann den Quell-Bucket Ihrer Replikationskonfiguration aus.
7. (Optional) Sie können zusätzliche Filter wie Objekt-Erstellungsdatum und Replikationsstatus einschließen. Beispiele zum Filtern nach Replikationsstatus finden Sie unter Angeben eines Manifests für einen Batch-Replikationsauftrag.
8. Um ein Manifest zu speichern, wählen Sie Save Batch Operations manifest (Manifest für Batchvorgänge speichern).

   a. Wenn Sie ein Manifest erstellen und speichern möchten, müssen Sie entweder Bucket in this account (Bucket in diesem Konto) oder Bucket in another AWS-Konto (Bucket in einem anderen) auswählen. Geben Sie den Bucket-Namen im Textfeld an.

   > **Note**
   >
   > Das generierte Manifest muss in derselben AWS-Region wie der Quell-Bucket gespeichert werden.

   b. Wählen Sie den Verschlüsselungstyp aus.
9. (Optional) Geben Sie eine Beschreibung ein.
10. Passen Sie die Priority (Priorität) des Auftrags bei Bedarf an. Höhere Nummern bedeuten eine höhere Priorität. Amazon S3 versucht, Aufträge mit höherer Priorität vor Aufträgen mit niedrigerer Priorität auszuführen. Weitere Informationen zur Auftragspriorität finden Sie unter Zuweisen der Auftragspriorität.
11. (Optional) Generieren Sie einen Abschlussbericht. Wählen Sie zum generieren Generate completion report (Abschlussbericht generieren).

    Wenn Sie einen Abschlussbericht erstellen möchten, müssen Sie entweder die Meldung von Failed tasks only (Nur fehlgeschlagenen Aufgaben) oder All tasks (Allen Aufgaben) auswählen und einen Ziel-Bucket für den Bericht angeben.

12. Wählen Sie eine gültige IAM-Rolle aus.

> **Note**
>
> Weitere Informationen zum Erstellen einer IAM-Rolle finden Sie unter Konfigurieren von IAM-Richtlinien für die Batch-Replikation.

13. (Optional) Fügen Sie Auftrags-Tags zum Batch-Replikationsauftrag hinzu.
14. Wählen Sie Weiter aus.
15. Überprüfen Sie Ihre Auftrags-Konfiguration und wählen Sie Create job (Auftrag erstellen).

Verwenden der AWS CLI mit einem S3-Manifest

Im folgenden Beispiel wird ein S3-Batch-Replikationsauftrag unter Verwendung eines S3-generierten Manifest für das AWS-Konto *111122223333* dargestellt. In diesem Beispiel wird versucht, vorhandene Objekte und Objekte, die zuvor nicht repliziert wurden, zu replizieren. Informationen zum Filtern nach Replikationsstatus finden Sie unter Angeben eines Manifests für einen Batch-Replikationsauftrag.

```
aws s3control create-job --account-id 111122223333 --operation
 '{"S3ReplicateObject":{}}' --report '{"Bucket":"arn:aws:s3:::***
 completion report bucket ****","Prefix":"batch-replication-report",
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}'
 --manifest-generator '{"S3JobManifestGenerator": {"ExpectedBucketOwner":
 "111122223333", "SourceBucket": "arn:aws:s3:::*** replication source bucket
 ***", "EnableManifestOutput": false, "Filter": {"EligibleForReplication": true,
 "ObjectReplicationStatuses": ["NONE","FAILED"]}}}' --priority 1 --role-arn
 arn:aws:iam::111122223333:role/batch-Replication-IAM-policy --no-confirmation-required
 --region source-bucket-region
```

> **Note**
>
> Der Auftrag muss aus demselben AWS-Region Replikationsquellen-Bucket initiiert werden. Die IAM-Rolle `role/batch-Replication-IAM-policy` wurde zuvor erstellt. Siehe Konfigurieren von IAM-Richtlinien für die Batch-Replikation.

Nachdem Sie einen Batch-Replikationsauftrag erfolgreich initiiert haben, erhalten Sie die Auftrags-ID als Antwort. Sie können diesen Auftrag mit dem folgenden Befehl überwachen.

```
aws s3control describe-job --account-id 111122223333 --job-id job-id --region source-
bucket-region
```

Verwenden der AWS CLI mit einem vom Benutzer bereitgestellten Manifest

Im folgenden Beispiel wird ein S3-Batch-Replikationsauftrag unter Verwendung eines benutzerdefinierten Manifest für das AWS-Konto *111122223333* erstellt. Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, müssen Sie die Versions-IDs für die Objekte angeben. Es wird nur das Objekt mit der im Manifest angegebenen Versions-ID repliziert. Weitere Informationen zum Erstellen eines Manifests finden Sie unter Angeben eines Manifests.

```
aws s3control create-job --account-id 111122223333 --operation
 '{"S3ReplicateObject":{}}' --report '{"Bucket":"arn:aws:s3:::***
 completion report bucket ****","Prefix":"batch-replication-report",
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}'
 --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key","VersionId"]},"Location":{"ObjectArn":"arn:aws:s3:::*** completion
 report bucket ****/manifest.csv","ETag":"Manifest Etag"}}' --priority 1 --role-arn
 arn:aws:iam::111122223333:role/batch-Replication-IAM-policy --no-confirmation-required
 --region source-bucket-region
```

> **Note**
>
> Der Auftrag muss aus demselben AWS-Region Replikationsquellen-Bucket initiiert werden. Die IAM-Rolle `role/batch-Replication-IAM-policy` wurde zuvor erstellt. Siehe Konfigurieren von IAM-Richtlinien für die Batch-Replikation.

Nachdem Sie einen Batch-Replikationsauftrag erfolgreich initiiert haben, erhalten Sie die Auftrags-ID als Antwort. Sie können diesen Auftrag mit dem folgenden Befehl überwachen.

```
aws s3control describe-job --account-id 111122223333 --job-id job-id --region source-
bucket-region
```

Zusätzliche Replikations-Konfigurationen

In diesem Abschnitt werden zusätzliche Optionen zur Replikations-Konfiguration beschrieben, die in Amazon S3 verfügbar sind. Weitere Informationen zum Erstellen einer grundlegenden Replikationskonfiguration finden Sie unter Einrichten der Replikation.

Themen

Überwachen des Fortschritts mit Replikationsmetriken und S3-Ereignisbenachrichtigungen

S3-Replikationsmetriken enthalten detaillierte Metriken für die Replikationsregeln in Ihrer Replikationskonfiguration. Mit Replikationsmetriken können Sie den minute-by-minute Fortschritt überwachen, indem Sie ausstehende Bytes, ausstehende Operationen, Operationen, bei denen die Replikation fehlgeschlagen ist, und die Replikationslatenz verfolgen.

S3-Replikationsmetriken werden automatisch aktiviert, wenn Sie die Funktion für die Begrenzung der S3-Replikationszeit (S3 RTC) aktivieren. Sie können S3-Replikationsmetriken auch unabhängig von S3 RTC aktivieren, während Sie eine Regel erstellen oder bearbeiten. S3 RTC umfasst weitere Merkmale wie ein Service Level Agreement (SLA) und Benachrichtigungen für nicht eingehaltene Schwellenwerte. Weitere Informationen finden Sie unter Erfüllen der Compliance-Anforderungen mit S3-Replikationszeitkontrolle (S3 RTC).

Die Metriken in Bezug auf ausstehende Bytes, ausstehende Operationen und Replikationslatenz gelten nur für neue Objekte, die mit der regionsübergreifenden S3-Replikation (S3 CRR) oder der regionsinternen S3-Replikation (S3 SRR) repliziert werden. Die Metrik für Operationen mit fehlgeschlagener Replikation verfolgt sowohl neue Objekte, die mit S3 CRR oder S3 SRR repliziert werden, als auch vorhandene Objekte, die mit S3-Batch-Replikation repliziert werden. Zur Unterstützung bei der Behebung von Konfigurationsproblemen können Sie zudem Amazon-S3-Ereignisbenachrichtigungen einrichten, die Sie über Replikations-Fehlerereignisse informieren.

Wenn diese Option aktiviert ist, veröffentlichen S3-Replikationsmetriken die folgenden Metriken in Amazon CloudWatch:

- Bytes der ausstehenden Replikation – Die Gesamtzahl der Bytes von Objekten, deren Replikation für eine bestimmte Replikationsregel aussteht.
- Replikationslatenz – Die maximale Anzahl von Sekunden, um die die Replikations-Ziel-Buckets für eine bestimmte Replikationsregel hinter dem Quell-Bucket zurückliegen.
- Operationen mit ausstehender Replikation – Die Anzahl der Operationen, deren Replikation für eine bestimmte Replikationsregel aussteht. Diese Metrik verfolgt Operationen im Zusammenhang mit Objekten, Löschmarkierungen, Tags, Zugriffssteuerungslisten (ACLs) und S3-Objektsperren.
- Operationen fehlgeschlagen Replikation – Die Anzahl der Operationen, deren Replikation für eine bestimmte Replikationsregel fehlgeschlagen ist. Diese Metrik verfolgt Operationen im Zusammenhang mit Objekten, Löschmarkierungen, Tags, ACLs und Objektsperren. Im Gegensatz zu den anderen Replikationsmetriken gilt diese Metrik sowohl für neue Objekte, die mit S3 CRR oder S3 SRR repliziert werden, als auch für vorhandene Objekte, die mit S3-Batch-Replikation repliziert werden.

> **Note**
>
> Operationen fehlgeschlagen Replikation verfolgt S3-Replikationsfehler zusammengefasst in einem minütlichen Intervall. Um die spezifischen Objekte zu ermitteln, bei denen die Replikation fehlgeschlagen ist, und die Fehlerursachen herauszufinden, abonnieren Sie das `OperationFailedReplication`-Ereignis in den Amazon-S3-Ereignisbenachrichtigungen. Weitere Informationen finden Sie unter Erhalten von Amazon-S3-Ereignisbenachrichtigungen über Replikations-Fehlerereignisse.

Wenn ein Auftrag überhaupt nicht ausgeführt werden kann, werden keine Metriken an Amazon gesendet CloudWatch. Ihr Auftrag wird beispielsweise nicht ausgeführt, wenn Sie nicht über die erforderlichen Berechtigungen verfügen, um einen S3-Batch-Replikationsauftrag auszuführen, oder wenn die Tags oder das Präfix in Ihrer Replikationskonfiguration nicht übereinstimmen.

Themen

- Aktivieren von S3-Replikationsmetriken
- Erhalten von Amazon-S3-Ereignisbenachrichtigungen über Replikations-Fehlerereignisse
- Anzeigen von Replikationsmetriken über die Amazon-S3-Konsole
- Gründe für das Fehlschlagen der Replikation in Amazon S3

Aktivieren von S3-Replikationsmetriken

Sie können S3-Replikationsmetriken mit einer neuen oder einer vorhandenen Replikationsregel verwenden. Sie können die Replikationsregel auf einen gesamten S3-Bucket oder auf Amazon-S3-Objekte mit einem bestimmten Präfix oder Tag anwenden.

Dieses Thema enthält Anleitungen zum Aktivieren von S3-Replikationsmetriken in Ihrer Replikationskonfiguration, wenn sich die Quell- und Ziel-Buckets im Eigentum desselben Kontos oder unterschiedlicher AWS-Konten befinden.

Um Replikationsmetriken mithilfe der AWS Command Line Interface (AWS CLI) zu aktivieren, müssen Sie dem Quell-Bucket eine Replikationskonfiguration hinzufügen, bei der `Metrics` aktiviert ist. In dieser Beispielkonfiguration werden Objekte unter dem Präfix *`Tax`* auf den Ziel-Bucket *`DOC-EXAMPLE-BUCKET`* repliziert und es werden Metriken für diese Objekte generiert.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "Metrics": {
                    "Status": "Enabled"
                }
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

Alle Informationen zum Erstellen von Replikationsregeln finden Sie unter Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos.

Weitere Informationen zum Anzeigen von Replikationsmetriken in der S3-Konsole finden Sie unter Anzeigen von Replikationsmetriken über die Amazon-S3-Konsole.

> **Note**
>
> S3-Replikationsmetriken werden zum gleichen Tarif wie CloudWatch benutzerdefinierte Amazon-Metriken abgerechnet. Weitere Informationen finden Sie unter Amazon CloudWatch-Preise.

Erhalten von Amazon-S3-Ereignisbenachrichtigungen über Replikations-Fehlerereignisse

S3-Ereignisbenachrichtigungen können Sie darüber informieren, wenn Objekte nicht in ihre Ziel-AWS-Region repliziert wurden. Amazon S3-Ereignisse sind über Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS) oder verfügbar AWS Lambda. Weitere Informationen finden Sie unter the section called “Amazon-S3-Ereignis-Benachrichtigungen”.

Eine Liste der Fehlercodes, die von S3-Ereignisbenachrichtigungen erfasst werden, finden Sie unter Gründe für das Fehlschlagen der Replikation in Amazon S3.

Anzeigen von Replikationsmetriken über die Amazon-S3-Konsole

Es gibt drei Arten von Amazon- CloudWatch Metriken für Amazon S3: Speichermetriken, Anforderungsmetriken und Replikationsmetriken. S3-Replikationsmetriken werden automatisch aktiviert, wenn Sie die Replikation mit S3 Replication Time Control (S3 RTC) mithilfe der AWS Management Console oder der Amazon S3-API aktivieren. Sie können S3-Replikationsmetriken auch unabhängig von S3 RTC aktivieren, während Sie eine Regel erstellen oder bearbeiten.

Replikationsmetriken verfolgen die Regel-IDs der Replikations-Konfiguration. Eine Replikationsregel-ID kann spezifisch für ein Präfix, ein Tag oder eine Kombination aus beiden sein.

Weitere Informationen zu CloudWatch Metriken für Amazon S3 finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

Voraussetzungen

Aktivieren Sie eine Replikationsregel, die über S3-Replikationsmetriken verfügt.

So zeigen Sie Replikationsmetriken an

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus. Wählen Sie in der Liste Buckets den Namen des Buckets mit den Objekten aus, für die Sie Replikationsmetriken abrufen möchten.

3. Wählen Sie den Tab Metrics.
4. Wählen Sie unter Replication metrics (Replikationsmetriken) die Option Replication rules (Replikationsregeln) aus.
5. Wählen Sie Display charts (Diagramme anzeigen).

   Amazon S3 zeigt die Diagramme Replikationslatenz (in Sekunden), Bytes der ausstehenden Replikation Operationen mit ausstehender Replikation und Operationen fehlgeschlagen Replikation an.

Sie können dann die Replikationsmetriken Replikationslatenz (in Sekunden), Operationen mit ausstehender Replikation, Bytes der ausstehenden Replikation und Operationen fehlgeschlagen Replikation für die ausgewählten Regeln anzeigen. Wenn Sie die S3-Replikationszeitkontrolle verwenden, CloudWatch beginnt Amazon mit der Meldung von Replikationsmetriken 15 Minuten, nachdem Sie S3 RTC für die jeweilige Replikationsregel aktiviert haben. Sie können Replikationsmetriken in der Amazon S3-Konsole oder der - CloudWatch Konsole anzeigen. Weitere Informationen finden Sie unter Replikationsmetriken mit S3 RTC.

Gründe für das Fehlschlagen der Replikation in Amazon S3

Die folgende Tabelle enthält Gründe für das Fehlschlagen der Replikation in Amazon S3. Sie können diese Gründe anzeigen, indem Sie das failureReason-Ereignis mit den Amazon-S3-Ereignisbenachrichtigungen erhalten. Sie können S3-Ereignisbenachrichtigungen über Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS) oder erhalten AWS Lambda. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Sie können diese Fehlergründe auch in einem Abschlussbericht der S3-Batch-Replikation anzeigen. Weitere Informationen finden Sie unter Abschlussbericht zur Batchreplikation.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`AssumeRoleNotPermitted`	Amazon S3 kann nicht die AWS Identity and Access Management (IAM)-Rolle übernehmen, die in der Replikationskonfiguration oder im Batchoperationenauftrag angegeben ist.
`DstBucketInvalidRegion`	Der Ziel-Bucket befindet sich nicht in derselben AWS-Region wie im Batchoper

Gründe für das Fehlschlagen der Replikation	Beschreibung
	ationenauftrag angegeben. Dieser Fehler ist spezifisch für die Batchreplikation.
`DstBucketNotFound`	Amazon S3 kann den in der Replikati onskonfiguration angegebenen Ziel-Bucket nicht finden.
`DstBucketObjectLockConfigMissing`	Um Objekte aus einem Quell-Bucket mit aktivierter Objektsperre zu replizieren, muss für den Ziel-Bucket ebenfalls die Objektspe rre aktiviert sein. Dieser Fehler zeigt an, dass die Objektsperre im Ziel-Bucket möglicher weise nicht aktiviert ist. Weitere Informationen finden Sie unter Überlegungen zu Object Lock.
`DstBucketUnversioned`	Die Versionsverwaltung ist für den S3-Ziel-Bucket nicht aktiviert. Aktivieren Sie die Versionsverwaltung für den Ziel-Bucket, um Objekte mit S3-Replikation zu replizieren.
`DstDelObjNotPermitted`	Amazon S3 kann Löschmarkierungen nicht in den angegebenen Ziel-Bucket replizier en. Möglicherweise fehlt die `s3:Replic ateDelete` -Berechtigung für den Ziel-Bucket.
`DstKmsKeyInvalidState`	Der AWS Key Management Service (AWS KMS)-Schlüssel für den Ziel-Bucket hat keinen gültigen Status. Überprüfen und aktivieren Sie den erforderlichen AWS KMS Schlüssel. Weitere Informationen zum Verwalten AWS KMS von Schlüsseln finden Sie unter Schlüsselstatus von AWS KMS Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`DstKmsKeyNotFound`	Der AWS KMS Schlüssel, der für den Ziel-Bucket in der Replikationskonfiguration konfiguriert ist, ist nicht vorhanden.
`DstMultipartCompleteNotPermitted`	Amazon S3 kann mehrteilige Uploads von Objekten im Ziel-Bucket nicht abschließ en. Möglicherweise fehlt die `s3:Replic ateObject` -Berechtigung für den Ziel-Bucket.
`DstMultipartInitNotPermitted`	Amazon S3 kann mehrteilige Uploads von Objekten in den Ziel-Bucket nicht initiiere n. Möglicherweise fehlt die `s3:Replic ateObject` -Berechtigung für den Ziel-Bucket.
`DstMultipartPartUploadNotPe rmitted`	Amazon S3 kann keine mehrteiligen Objekte in den Ziel-Bucket hochladen. Möglicherweise fehlt die `s3:ReplicateObject` -Berechti gung für den Ziel-Bucket.
`DstObjectHardDeleted`	Die S3-Batchreplikation unterstützt nicht das erneute Replizieren von Objekten, die mit der Versions-ID des Objekts aus dem Ziel-Bucket gelöscht wurden. Dieser Fehler ist spezifisch für die Batchreplikation.
`DstPutAclNotPermitted`	Amazon S3 kann Objekt-Zugriffssteuerungsli sten (ACLs) nicht in den angegebenen Ziel-Bucket replizieren. Möglicherweise fehlt die `s3:ReplicateObject` -Berechtigung für den Ziel-Bucket.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`DstPutLegalHoldNotPermitted`	Amazon S3 kann keine Objektsperre aufgrund gesetzlicher Aufbewahrungsfristen für die Zielobjekte festlegen, während unveränderliche Objekte repliziert werden. Möglicherweise fehlt die `s3:PutObjectLegalHold` -Berechtigung für den Ziel-Bucket. Weitere Informationen finden Sie unter Rechtliche Aufbewahrungsfristen.
`DstPutObjectNotPermitted`	Amazon S3 kann Objekte nicht in den angegebenen Ziel-Bucket replizieren. Möglicherweise fehlen die `s3:ReplicateObject` - oder `s3:ObjectOwnerOverrideToBucketOwner` -Berechtigungen für den Ziel-Bucket.
`DstPutTaggingNotPermitted`	Amazon S3 kann Objekt-Tags nicht in den angegebenen Ziel-Bucket replizieren. Möglicherweise fehlt die `s3:ReplicateObject` -Berechtigung für den Ziel-Bucket.
`DstVersionNotFound`	Amazon S3 kann die erforderliche Objektversion im Ziel-Bucket nicht finden, für den Metadaten repliziert werden müssen.
`InitiateReplicationNotPermitted`	Amazon S3 kann keine Replikation für Objekte initiieren. Möglicherweise fehlt die `s3:InitiateReplication` -Berechtigung für den Batch-Operationsauftrag. Dieser Fehler ist spezifisch für die Batchreplikation.
`SrcBucketInvalidRegion`	Der Quell-Bucket befindet sich nicht in derselben AWS-Region wie im Batchoperationenauftrag angegeben. Dieser Fehler ist spezifisch für die Batchreplikation.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`SrcBucketNotFound`	Amazon S3 kann den Quell-Bucket nicht finden.
`SrcBucketReplicationConfigMissing`	Amazon S3 konnte keine Replikationskonfiguration für den Quell-Bucket finden.
`SrcGetAclNotPermitted`	Amazon S3 kann nicht auf das Objekt im Quell-Bucket für die Replikation zugreifen. Möglicherweise fehlt die `s3:GetObjectVersionAcl` -Berechtigung für das Quell-Bucket-Objekt. Die Objekte im Quell-Bucket müssen sich im Besitz des Bucket-Eigentümers befinden. Wenn ACLs aktiviert sind, überprüfen Sie, ob die Objekteigentümerschaft auf Bucket-Eigentümer bevorzugt oder auf Objektschreiber festgelegt ist. Wenn die Objekteigentümerschaft auf Bucket-Eigentümer bevorzugt festgelegt ist, müssen die Quell-Bucket-Objekte über die ACL `bucket-owner-full-control` verfügen, damit der Bucket-Eigentümer zum Objekteigentümer wird. Das Quellkonto kann die Eigentümerschaft für alle Objekte in seinem Bucket übernehmen, indem die Objekteigentümerschaft auf Bucket-Eigentümer erzwungen festgelegt und die ACLs deaktiviert werden.
`SrcGetLegalHoldNotPermitted`	Amazon S3 kann nicht auf die Informationen zur gesetzlichen Aufbewahrung von S3 Object Lock zugreifen.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`SrcGetObjectNotPermitted`	Amazon S3 kann nicht auf das Objekt im Quell-Bucket für die Replikation zugreifen . Möglicherweise fehlt die `s3:GetObjectVersionForReplication` -Berechtigung für den Quell-Bucket.
`SrcGetRetentionNotPermitted`	Amazon S3 kann nicht auf die Informationen zum Aufbewahrungszeitraum von S3 Object Lock zugreifen.
`SrcGetTaggingNotPermitted`	Amazon S3 kann nicht auf Objekt-Tag-Informationen aus dem Quell-Bucket zugreifen. Möglicherweise fehlt die `s3:GetObjectVersionTagging` -Berechtigung für den Quell-Bucket.
`SrcHeadObjectNotPermitted`	Amazon S3 kann keine Objektmetadaten aus dem Quell-Bucket abrufen. Möglicherweise fehlt die `s3:GetObjectVersionForReplication` -Berechtigung für den Quell-Bucket.
`SrcKeyNotFound`	Amazon S3 kann den Quellobjektschlüssel für die Replikation nicht finden. Das Quellobjekt wurde möglicherweise gelöscht, bevor die Replikation abgeschlossen war.
`SrcKmsKeyInvalidState`	Der AWS KMS Schlüssel für den Quell-Bucket befindet sich nicht in einem gültigen Zustand. Überprüfen und aktivieren Sie den erforderlichen AWS KMS Schlüssel. Weitere Informationen zum Verwalten von AWS KMS Schlüsseln finden Sie unter Schlüsselstatus von AWS KMS Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`SrcObjectNotEligible`	Einige Objekte kommen nicht für die Replikation infrage. Dies kann an der Speicherklasse des Objekts liegen oder daran, dass die Objekt-Tags nicht mit der Replikationskonfiguration übereinstimmen.
`SrcObjectNotFound`	Das Quellobjekt ist nicht vorhanden.
`SrcReplicationNotPending`	Amazon S3 hat dieses Objekt bereits repliziert. Für dieses Objekt steht keine Replikation mehr aus.
`SrcVersionNotFound`	Amazon S3 kann den Quellobjektversion für die Replikation nicht finden. Die Quellobjektversion wurde möglicherweise gelöscht, bevor die Replikation abgeschlossen war.

Verwandte Themen

Einrichten von Berechtigungen

Fehlerbehebung bei einer Replikation

Erfüllen der Compliance-Anforderungen mit S3-Replikationszeitkontrolle (S3 RTC)

S3 Replication Time Control (S3 RTC) hilft Ihnen bei der Einhaltung von Compliance- oder Geschäftsanforderungen für die Datenreplikation und bietet Einblick in die Amazon-S3-Replikationsaktivitäten. S3 RTC repliziert die meisten Objekte, die Sie zu Amazon S3 hochladen, in Sekunden und 99,99 Prozent dieser Objekte innerhalb von 15 Minuten.

Die S3 RTC umfasst standardmäßig S3-Replikationsmetriken und S3-Ereignis-Benachrichtigungen, mit denen Sie die Gesamtzahl der S3-API-Operationen, die zur Replikation anstehen, die Gesamtgröße der zur Replikation anstehenden Objekte und die maximale Replikationszeit überwachen können. Replikationsmetriken können unabhängig von S3 RTC aktiviert werden, siehe Fortschritt mit Replikationsmetriken überwachen. Darüber hinaus bietet S3 RTC `OperationMissedThreshold`- und `OperationReplicatedAfterThreshold`-Ereignisse,

die den Bucket-Besitzer benachrichtigen, wenn die Objektreplikation den 15-Minuten-Grenzwert überschreitet oder danach repliziert wird.

Mit S3 RTC können Sie in den seltenen Fällen, in denen Objekte nicht innerhalb von 15 Minuten repliziert werden und diese Objekte nach dem Schwellenwert von 15 Minuten repliziert werden, Amazon-S3-Ereignisse erhalten. Amazon S3-Ereignisse sind über Amazon SQS , Amazon SNS oder verfügbar AWS Lambda. Weitere Informationen finden Sie unter the section called “Amazon-S3-Ereignis-Benachrichtigungen”.

Themen

- Aktivieren der S3 Replication Time Control
- Replikationsmetriken mit S3 RTC
- Verwenden von Amazon-S3-Ereignis-Benachrichtigungen zum Nachverfolgen von Replikationsobjekten
- Bewährte Methoden und Richtlinien für S3 RTC

Aktivieren der S3 Replication Time Control

Sie können die S3-Replikationszeitkontrolle (S3 RTC) mit einer neuen oder einer vorhandenen Replikationsregel verwenden. Sie können die Replikationsregel auf einen gesamten S3-Bucket oder auf Amazon-S3-Objekte mit einem bestimmten Präfix oder Tag anwenden. Wenn Sie S3 RTC aktivieren, werden Replikationsmetriken auch für Ihre Replikationsregel aktiviert.

Wenn Sie die aktuelle Version der Replikations-Konfiguration verwenden, d. h. das Element `Filter` in einer Replikations-Konfigurations-Regel angeben, repliziert Amazon S3 die Löschmarkierung nicht standardmäßig. Sie können Regeln jedoch die Replikation von non-tag-based Löschmarkierungen hinzufügen.

> **Note**
>
> Replikationsmetriken werden zum gleichen Tarif wie CloudWatch benutzerdefinierte Amazon-Metriken abgerechnet. Weitere Informationen finden Sie unter Amazon CloudWatch-Preise.

Weitere Informationen zum Erstellen einer Regel mit S3 RTC finden Sie unter Replizieren von Objekten mit S3-Replikationszeitkontrolle (S3 RTC).

Replikationsmetriken mit S3 RTC

Replikationsregeln mit aktivierter S3-Replikationszeitkontrolle (S3 RTC) veröffentlichen Replikationsmetriken. Mit Replikationsmetriken können Sie die Gesamtzahl der S3-API-Operationen mit ausstehender Replikation, die Gesamtgröße der Objekte mit ausstehender Replikation, die maximale Replikationszeit in der Zielregion und der Gesamtzahl der Operationen überwachen, deren Replikation fehlgeschlagen ist. Anschließend können Sie jedes Dataset, das Sie separat replizieren, überwachen.

Replikationsmetriken sind innerhalb von 15 Minuten nach der Aktivierung von S3 RTC verfügbar. Replikationsmetriken sind über die Amazon S3-Konsole, die Amazon S3-API, die - AWS SDKs, die AWS Command Line Interface (AWS CLI) und Amazon CloudWatch verfügbar. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

Weitere Informationen zum Finden von Replikationsmetriken über die Amazon-S3-Konsole finden Sie unter Anzeigen von Replikationsmetriken über die Amazon-S3-Konsole.

Verwenden von Amazon-S3-Ereignis-Benachrichtigungen zum Nachverfolgen von Replikationsobjekten

Sie können die Replikationszeit für Objekte nachverfolgen, die nicht innerhalb von 15 Minuten repliziert wurden, indem Sie bestimmte Ereignis-Benachrichtigungen überwachen, die von der S3-Replikationszeitkontrolle (S3 RTC) veröffentlicht werden. Diese Ereignisse werden veröffentlicht, wenn ein Objekt, das für die Replikation mit S3 RTC in Frage kam, nicht innerhalb von 15 Minuten repliziert wurde und wenn dieses Objekt nach dem Schwellenwert von 15 Minuten repliziert wird.

Replikationsmetriken sind innerhalb von 15 Minuten nach der Aktivierung von S3 RTC verfügbar. Amazon S3-Ereignisse sind über Amazon SQS , Amazon SNS oder verfügbar AWS Lambda. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Bewährte Methoden und Richtlinien für S3 RTC

Befolgen Sie beim Replizieren von Daten in Amazon S3 mit der S3-Replikationszeitkontrolle (S3 RTC) diese bewährten Methoden, um die Replikationsleistung für Ihre Workloads zu optimieren.

Themen

- Leistungsrichtlinien für Amazon-S3-Replikation and -Anforderungsraten
- Schätzen der Replikationsanforderungsraten
- Überschreiten der Grenzwerte für die S3 RTC-Datenübertragungsrate

- AWS KMS Replikationsanforderungsraten für verschlüsselte Objekte

Leistungsrichtlinien für Amazon-S3-Replikation and -Anforderungsraten

Wenn Speicherinhalte zu Amazon S3 hochgeladen oder von dort abgerufen werden, können Ihre Anwendungen Tausende von Transaktionen pro Sekunde bei der Anforderungsleistung erhalten. Beispielsweise kann eine Anwendung mindestens 3 500 PUT/COPY/POST/DELETE- oder 5 500 GET/HEAD-Anforderungen pro Sekunde pro Präfix in einem S3-Bucket erreichen, einschließlich der Anforderungen, die die S3-Replikation in Ihrem Namen vornimmt. Es gibt keine Einschränkungen für die Anzahl der Präfixe in einem Bucket. Sie können Ihre Lese- und Schreibleistung steigern, indem Sie Lesevorgänge parallelisieren. Wenn Sie beispielsweise 10 Präfixe in einem S3-Bucket für parallele Lesevorgänge einrichten, können Sie damit die Leseleistung auf 55 000 Leseanfragen pro Sekunde skalieren.

Amazon S3 skaliert automatisch als Reaktion auf anhaltende Anforderungsraten oberhalb dieser Richtlinien oder anhaltender Anforderungsraten übereinstimmend mit LIST-Anforderungen. Während Amazon S3 intern für die neue Anforderungsrate optimiert wird, erhalten Sie möglicherweise temporär HTTP 503-Anforderungsantworten, bis die Optimierung abgeschlossen ist. Dies kann mit steigenden Anforderungsraten pro Sekunde oder beim ersten Aktivieren von S3 RTC auftreten. Während dieser Zeiträume kann sich die Replikationslatenz erhöhen. Das S3 RTC Service Level Agreement (SLA) gilt nicht für Zeiträume, in denen Amazon-S3-Leistungsrichtlinien für Anforderungen pro Sekunde überschritten werden.

Das S3 RTC SLA gilt auch nicht in Zeiträumen, in denen Ihre Replikationsdatenübertragungsrate das Standardlimit von 1 Gbit/s überschreitet. Wenn Sie erwarten, dass Ihre Replikations-Übertragungsrate 1 Gbit/s überschreitet, können Sie Ihr AWS Support -Center kontaktieren oder über Service Quotas eine Erhöhung Ihres Limit anzufordern.

Schätzen der Replikationsanforderungsraten

Ihre Gesamtanforderungsrate einschließlich der Anforderungen, die die Amazon-S3-Replikation in Ihrem Namen vornimmt, sollte den Richtlinien für die Amazon-S3-Anforderungsrate sowohl für die Replikationsquelle als auch für die Ziel-Buckets entsprechen. Für jedes replizierte Objekt führt die Amazon-S3-Replikation bis zu fünf GET/HEAD-Anforderungen und eine PUT-Anforderung an den Quell-Bucket sowie eine PUT-Anforderung an jeden Ziel-Bucket aus.

Wenn Sie beispielsweise davon ausgehen, dass 100 Objekte pro Sekunde repliziert werden, kann die Amazon-S3-Replikation für Sie zusätzliche 100 PUT-Anfragen für insgesamt 200 PUT-pro-Sekunden

in den Quell-S3-Bucket ausführen. Die Amazon-S3-Replikation kann außerdem bis zu 500 GET/HEAD-Anforderungen ausführen (5 GET/HEAD-Anforderungen für jedes replizierte Objekt)

> **Note**
>
> Es entstehen Kosten für nur eine PUT-Anforderung pro repliziertem Objekt. Weitere Informationen finden Sie in den Preisinformationen unter Amazon S3 – Häufig gestellte Fragen zur Replikation.

Überschreiten der Grenzwerte für die S3 RTC-Datenübertragungsrate

Wenn Sie erwarten, dass die Datenübertragungsrate von S3 Replication Time Control das Standardlimit von 1 Gbit/s überschreitet, wenden Sie sich an Ihr AWS Support -Center oder verwenden Sie Service Quotas, um eine Erhöhung Ihres Limit anzufordern.

AWS KMS Replikationsanforderungsraten für verschlüsselte Objekte

Wenn Sie Objekte replizieren, die mit serverseitiger Verschlüsselung (SSE-KMS) mit Amazon S3-Replikation verschlüsselt sind, gelten AWS Key Management Service (AWS KMS)-Anforderungen pro Sekunde Limits. AWS KMS lehnt möglicherweise eine ansonsten gültige Anforderung ab, da Ihre Anforderungsrate das Limit für die Anzahl der Anforderungen pro Sekunde überschreitet. Wenn eine Anforderung gedrosselt wird, AWS KMS gibt einen `ThrottlingException` Fehler zurück. Das AWS KMS Anforderungsratenlimit gilt für Anforderungen, die Sie direkt stellen, und für Anforderungen, die von der Amazon S3-Replikation in Ihrem Namen gestellt werden.

Wenn Sie beispielsweise erwarten, 1 000 Objekte pro Sekunde zu replizieren, können Sie 2 000 Anfragen von Ihrem AWS KMS Anforderungsratenlimit subtrahieren. Die resultierende Anforderungsrate pro Sekunde ist für Ihre AWS KMS Workloads ohne Replikation verfügbar. Sie können AWS KMS Anforderungsmetriken in Amazon CloudWatch verwenden, um die AWS KMS Gesamtanforderungsrate in Ihrem zu überwachen AWS-Konto.

Replizieren von Löschmarkierungen auf Buckets

Wenn S3-Replikation aktiviert ist und ein Objekt im Quell-Bucket gelöscht wird, fügt Amazon S3 standardmäßig nur im Quell-Bucket eine Löschmarkierung hinzu. Dies schützt Daten vor missbräuchlichen Löschungen.

Wenn Sie die Löschmarkierungsreplikation aktiviert haben, werden diese Markierungen in die Ziel-Buckets kopiert und Amazon S3 verhält sich so, als sei das Objekt sowohl in den Quell-

als auch in den Ziel-Buckets gelöscht worden. Weitere Informationen zur Funktionsweise von Löschmarkierungen finden Sie unter Arbeiten mit Löschmarkierungen.

> **Note**
>
> Die Replikation von Löschmarkierungen wird für Tag-basierte Replikationsregeln nicht unterstützt. Die Löschmarkierungs-Replikation erfüllt auch nicht das SLA von 15 Minuten, die bei Verwendung der S3-Replikationszeitkontrolle gewährt werden.

Wenn Sie nicht die neueste Version der Replikations-Konfiguration verwenden, wirken sich Löschvorgänge unterschiedlich auf die Replikation aus. Weitere Informationen finden Sie unter Auswirkungen von LöschVorgänge auf die Replikation.

Aktivieren der Löschmarkierungs-Replikation

Sie können die Löschmarkierungs-Replikation mit einer neuen oder einer vorhandenen Replikationsregel verwenden. Sie können sie auf einen gesamten S3-Bucket oder auf Amazon-S3-Objekte anwenden, die ein bestimmtes Präfix haben.

Informationen zum Aktivieren der Löschung von Markierungsreplikationen mit der Amazon-S3-Konsole finden Sie unter Verwenden der S3-Konsole. Dieses Thema enthält Anweisungen zum Aktivieren der Löschmarkierungsreplikation in Ihrer Replikationskonfiguration, wenn sich Buckets im Besitz desselben oder unterschiedlicher befinden AWS-Konten.

Um die Löschmarkierungsreplikation mit der AWS Command Line Interface (AWS CLI) zu aktivieren, müssen Sie dem Quell-Bucket eine Replikationskonfiguration hinzufügen, bei der `DeleteMarkerReplication` aktiviert ist.

In der folgenden Beispielkonfiguration werden Löschmarkierungen für Objekte unter dem Präfix *`Tax`* in den Ziel-Bucket *`DOC-EXAMPLE-BUCKET`* repliziert.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "DeleteMarkerReplication": {
```

```
                "Status": "Enabled"
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

Vollständige Anweisungen zum Erstellen von Replikationsregeln über die finden Sie AWS CLI unter Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos im Abschnitt Replikations-Walkthroughs.

Replizieren von Metadatenänderungen mit der Synchronisierung von Amazon-S3-Replikatänderungen

Die Synchronisierung von Amazon-S3-Replikatänderungen kann Ihnen helfen, Objektmetadaten wie Markierungen, ACLs und Objektsperren-Einstellungen zwischen Replikaten und Quellobjekten zu replizieren. Standardmäßig repliziert Amazon S3 Metadaten aus den Quellobjekten nur auf die Replikate. Wenn die Synchronisierung von Replikatänderungen aktiviert ist, repliziert Amazon S3 Metadatenänderungen an den Replikatkopien zurück auf das Quellobjekt, sodass die Replikation bidirektional wird.

Aktivieren der Synchronisierung von Replikatänderungen

Sie können die Synchronisierung von Amazon-S3-Replikatänderungen mit neuen oder vorhandenen Replikationsregeln verwenden. Sie können sie auf einen gesamten S3-Bucket oder auf Amazon-S3-Objekte anwenden, die ein bestimmtes Präfix haben.

Informationen zum Aktivieren der Synchronisierung von Replikatänderungen über die Amazon-S3-Konsole finden Sie unter Anleitungen: Beispiele zum Konfigurieren der Replikation. Dieses Thema enthält Anweisungen zum Aktivieren der Synchronisierung von Replikatänderungen in Ihrer Replikationskonfiguration, wenn sich Buckets im Besitz desselben oder unterschiedlicher befinden AWS-Konten.

Um die Synchronisierung von Replikatänderungen mit der AWS Command Line Interface (AWS CLI) zu aktivieren, müssen Sie dem Bucket, der die Replikate enthält, eine Replikationskonfiguration hinzufügen, bei der `ReplicaModifications` aktiviert ist. Um eine bidirektionale Replikation

einzurichten, erstellen Sie eine Replikationsregel vom Quell-Bucket (*DOC-EXAMPLE-BUCKET1*) zu dem Bucket, der die Replikate enthält (*DOC-EXAMPLE-BUCKET2*). Erstellen Sie dann eine zweite Replikationsregel von dem Bucket, der die Replikate enthält (*DOC-EXAMPLE-BUCKET2*), zum Quell-Bucket (*DOC-EXAMPLE-BUCKET1*). Buckets können sich im selben oder in unterschiedlichen befinden AWS-Regionen.

> **Note**
>
> Sie müssen die Synchronisierung von Replikatänderungen in beiden Buckets aktivieren, um Änderungen an Replikatmetadaten wie Objektzugriffssteuerungslisten (ACLs), Objekttags oder Objektsperreinstellungen auf den replizierten Objekten zu replizieren. Wie alle Replikationsregeln können diese Regeln entweder auf den gesamten Amazon-S3-Bucket oder auf eine nach Präfix oder Objekt-Tags gefilterte Teilmenge von Amazon-S3-Objekten angewendet werden.

In der folgenden Beispielkonfiguration repliziert Amazon S3 Metadatenänderungen unter dem Präfix *Tax* auf den Bucket *DOC-EXAMPLE-BUCKET*, der die Quellobjekte enthält.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "SourceSelectionCriteria": {
                "ReplicaModifications":{
                    "Status": "Enabled"
                }
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

Vollständige Anweisungen zum Erstellen von Replikationsregeln mit der finden Sie AWS CLI unter Konfigurieren der Replikation für Quell- und Ziel-Buckets im Eigentum desselben Kontos.

Ändern des Replikat-Eigentümers

Bei der Replikation besitzt der Eigentümer des Quellobjekts standardmäßig auch das Replikat. Wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten und Sie die Replikateigentümerschaft in den ändern möchten AWS-Konto , der die Ziel-Buckets besitzt, können Sie optionale Konfigurationseinstellungen hinzufügen, um die Replikateigentümerschaft in den zu ändern AWS-Konto , der die Ziel-Buckets besitzt. Sie können dies z. B. tun, um den Zugriff auf Objektreplikate einzuschränken. Dies wird auch als die Eigentümer-Überschreibungs-Option der Replikationskonfiguration bezeichnet. Weitere Informationen zur Besitzer-Überschreibungs-Option finden Sie unter Hinzufügen der Eigentümer-Überschreibungs-Option zur Replikations-Konfiguration. Weitere Informationen zum Einrichten der Replikationskonfiguration finden Sie unter Replizieren von Objekten.

Um die Eigentümer-Überschreibung zu konfigurieren, gehen Sie wie folgt vor:

- Fügen Sie die Eigentümer-Überschreibungs-Option zur Replikations-Konfiguration hinzu, um Amazon S3 anzuweisen, die Replikat-Eigentümerschaft zu ändern.
- Erteilen Sie Amazon-S3-Berechtigungen zum Ändern der Replikat-Eigentümerschaft.
- Fügen Sie in der Richtlinie für die Ziel-Buckets die Berechtigung zum Ändern der Replikat-Eigentümerschaft hinzu. So kann der Eigentümer der Ziel-Buckets die Eigentümerschaft von Objektreplikaten annehmen.

Weitere Informationen finden Sie unter Hinzufügen der Eigentümer-Überschreibungs-Option zur Replikations-Konfiguration. Ein funktionierendes Beispiel mit step-by-step Anweisungen finden Sie unter Ändern des Replikat-Eigentümers, wenn sich Quell- und Ziel-Buckets im Eigentum unterschiedlicher Konten befinden.

Vom Bucket-Eigentümer erzwungene Einstellung für Object Ownership

Wenn Sie die Amazon S3-Replikation verwenden und sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten, kann der Bucket-Eigentümer des Ziel-Buckets ACLs deaktivieren (mit der Einstellung „Bucket-Eigentümer erzwungen“ für Object Ownership), um die Replikateigentümerschaft in den zu ändern AWS-Konto , der den Ziel-Bucket besitzt. Diese Einstellung ahmt das Verhalten der bestehenden Besitzerüberschreibung nach, ohne dass eine `s3:ObjectOwnerOverrideToBucketOwner`-Berechtigung erforderlich ist. Dies bedeutet, dass alle

Objekte, die mit der erzwungenen Einstellung des Bucket-Eigentümers in den Ziel-Bucket repliziert werden, dem Eigentümer des Ziel-Buckets gehören. Informationen zu Object Ownership finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Hinzufügen der Eigentümer-Überschreibungs-Option zur Replikations-Konfiguration

> **Warning**
>
> Fügen Sie die Eigentümer-Überschreibungsoption nur hinzu, wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten. Amazon S3 überprüft nicht, ob die Buckets im Besitz von gleichen oder unterschiedlichen Konten sind. Wenn Sie die Eigentümer-Überschreibung hinzufügen, wenn sich beide Buckets im Besitz desselben befinden AWS-Konto, wendet Amazon S3 die Eigentümer-Überschreibung an. Sie gewährt dem Ziel-Bucket-Eigentümer vollständige Berechtigungen und repliziert keine nachfolgenden Aktualisierungen der Quellobjekt-Access-Control-List (ACL). Der Replikateigentümer kann die ACL, die mit einem Replikat mit einer `PUT ACL`-Anforderung verknüpft ist, direkt ändern, aber nicht über eine Replikation.

Um die Option zur Eigentümer-Überschreibung festzulegen, fügen Sie Folgendes zu jedem `Destination`-Element hinzu:

- Das Element `AccessControlTranslation`, das Amazon S3 anweist, die Replikateigentümerschaft zu ändern
- Das -`Account`Element, das die AWS-Konto des Ziel-Bucket-Eigentümers angibt

```
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    ...
    <Destination>
      ...
      <AccessControlTranslation>
           <Owner>Destination</Owner>
       </AccessControlTranslation>
      <Account>destination-bucket-owner-account-id</Account>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Die folgende Beispiel-Replikationskonfiguration weist Amazon S3 an, Objekte mit dem Schlüsselpräfix `Tax` in den Ziel-Bucket zu replizieren und die Replikateigentümerschaft zu ändern.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Role>arn:aws:iam::account-id:role/role-name</Role>
   <Rule>
      <ID>Rule-1</ID>
      <Priority>1</Priority>
      <Status>Enabled</Status>
      <DeleteMarkerReplication>
         <Status>Disabled</Status>
      </DeleteMarkerReplication>
      <Filter>
         <Prefix>Tax</Prefix>
      </Filter>
      <Destination>
         <Bucket>arn:aws:s3:::destination-bucket</Bucket>
         <Account>destination-bucket-owner-account-id</Account>
         <AccessControlTranslation>
            <Owner>Destination</Owner>
         </AccessControlTranslation>
      </Destination>
   </Rule>
</ReplicationConfiguration>
```

Erteilen der Berechtigung zur Änderung der Replikat-Eigentümerschaft an Amazon S3

Erteilen Sie Amazon S3 die Berechtigungen zum Ändern der Replikateigentümerschaft, indem Sie die Berechtigung für die Aktion `s3:ObjectOwnerOverrideToBucketOwner` zur Berechtigungsrichtlinie hinzufügen, die mit der IAM-Rolle verknüpft ist. Dies ist die IAM-Rolle, die Sie in der Replikations-Konfiguration festgelegt haben und die es Amazon S3 gestattet, Objekte in Ihrem Namen anzunehmen und zu replizieren.

```
...
{
    "Effect":"Allow",
         "Action":[
       "s3:ObjectOwnerOverrideToBucketOwner"
    ],
    "Resource":"arn:aws:s3:::destination-bucket/*"
}
```

```
...
```

Hinzufügen der Berechtigung zur Ziel-Bucket-Richtlinie, um das Ändern der Replikat-Eigentümerschaft zuzulassen

Der Eigentümer des Ziel-Buckets muss dem Eigentümer des Quell-Buckets die Berechtigung zum Ändern der Replikat-Eigentümerschaft erteilen. Der Eigentümer des Ziel-Buckets erteilt dem Eigentümer des Quell-Buckets die Berechtigung für die Aktion `s3:ObjectOwnerOverrideToBucketOwner`. Dies ermöglicht dem Eigentümer des Ziel-Buckets, die Eigentümerschaft von Objektreplikaten anzunehmen. Die folgende Beispielanweisung einer Bucket-Richtlinie zeigt, wie dies funktioniert:

```
...
{
    "Sid":"1",
    "Effect":"Allow",
    "Principal":{"AWS":"source-bucket-account-id"},
    "Action":["s3:ObjectOwnerOverrideToBucketOwner"],
    "Resource":"arn:aws:s3:::destination-bucket/*"
}
...
```

Weitere Überlegungen

Wenn Sie die Eigentümer-Überschreibungs-Option konfigurieren, berücksichtigen Sie die folgenden Überlegungen:

- Standardmäßig besitzt der Eigentümer des Quellobjekts auch das Replikat. Amazon S3 repliziert die Objektversion und die damit verbundene ACL.

  Wenn Sie die Eigentümer-Überschreibung hinzufügen, repliziert Amazon S3 nur die Objektversion, nicht die ACL. Darüber hinaus repliziert Amazon S3 keine nachfolgenden Änderungen an der ACL des Quellobjekts. Amazon S3 legt die ACL für das Replikat fest, das dem Ziel-Bucket-Eigentümer Vollzugriff erteilt.
- Wenn Sie eine Replikations-Konfiguration ändern und die Eigentümerüberschreibung aktivieren oder deaktivieren, geschieht Folgendes:

  - Wenn Sie die Eigentümerüberschreibungs-Option zur Replikations-Konfiguration hinzufügen

Wenn Amazon S3 eine Objektversion repliziert, verwirft es die ACL, die mit dem Quellobjekt verknüpft ist. Es legt stattdessen die ACL für das Replikat fest, sodass der Ziel-Bucket-Eigentümer vollständige Kontrolle erhält. Es repliziert keine nachfolgenden Änderungen an der Quellobjekt-ACL. Diese Änderung der ACL gilt nicht für Objektversionen, die repliziert wurden, bevor Sie die Eigentümer-Überschreibungs-Option festgelegt haben. ACL-Aktualisierungen an den Quellobjekten, die repliziert wurden, bevor die Eigentümer-Überschreibungs-Option festgelegt wurde, werden weiterhin repliziert (da das Objekt und seine Replikate weiterhin denselben Eigentümer haben).

- Wenn Sie die Eigentümer-Überschreibungs-Option aus der Replikations-Konfiguration entfernen

  Amazon S3 repliziert neue Objekte, die im Quell-Bucket erscheinen, und die zugehörigen ACLs in die Ziel-Buckets. Bei Objekten, die repliziert wurden, bevor Sie die Eigentümer-Überschreibung entfernt haben, repliziert Amazon S3 die ACLs nicht, da die Änderung der Objekteigentümerschaft, die Amazon S3 vorgenommen hat, gültig bleibt. Das bedeutet, dass in der Objektversion abgelegte ACLs, die repliziert wurden, als Sie die Eigentümer-Überschreibungs-Option festgelegt hatten, weiterhin nicht repliziert werden.

Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Bei der Replikation von Objekten, die mit der serverseitigen Verschlüsselung verschlüsselt wurden, sind einige besondere Punkte zu beachten. Amazon S3 unterstützt die folgenden Arten von serverseitiger Verschlüsselung:

- Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)
- Serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS)
- Serverseitige Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS)
- Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter the section called “Server-side encryption”.

In diesem Thema werden die Berechtigungen erläutert, die Sie benötigen, um Amazon S3 anzuweisen, Objekte zu replizieren, die mithilfe serverseitiger Verschlüsselung verschlüsselt wurden. Dieses Thema enthält auch zusätzliche Konfigurationselemente, die Sie hinzufügen können, sowie Beispiele für AWS Identity and Access Management (IAM)-Richtlinien, die die erforderlichen Berechtigungen für die Replikation verschlüsselter Objekte gewähren.

Ein Beispiel mit step-by-step Anweisungen finden Sie unter Replizieren verschlüsselter Objekte. Weitere Informationen zum Erstellen einer Replikationskonfiguration finden Sie unter Replizieren von Objekten.

> **Note**
>
> Sie können Multi-Region AWS KMS keys in Amazon S3 verwenden. Amazon S3 behandelt jedoch derzeit Multi-Regions-Schlüssel wie Einzel-Regions-Schlüssel und verwendet nicht die Multi-Regions-Funktionen des Schlüssels. Weitere Informationen finden Sie unter Using multi-Region keys (Verwenden von Multi-Regions-Zugriffpunkt-Schlüsseln) im AWS Key Management Service -Entwicklerhandbuch.

Themen

- So wirkt sich die Standard-Bucket-Verschlüsselung auf die Replikation aus
- Mit SSE-C verschlüsselte Objekte replizieren
- Replizieren von mit SSE-S3, SSE-KMS oder DSSE-KMS verschlüsselten Objekten

So wirkt sich die Standard-Bucket-Verschlüsselung auf die Replikation aus

Wenn Sie die Standard-Verschlüsselung für einen Replikations-Ziel-Bucket aktivieren, gilt das folgende Verschlüsselungsverhalten:

- Wenn Objekte im Quell-Bucket nicht verschlüsselt sind, werden die Replikatobjekte im Ziel-Bucket mithilfe der Einstellungen der Standard-Verschlüsselung des Ziel-Buckets verschlüsselt. Daher unterscheiden sich die ETags (Entity-Tags) der Quellobjekte von den ETags der Replikatobjekte. Wenn Sie Anwendungen haben, die ETags verwenden, müssen Sie diese Anwendungen aktualisieren, um diesen Unterschied auszugleichen.
- Wenn Objekte im Quell-Bucket mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3), serverseitiger Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder serverseitiger Dual-Layer-Verschlüsselung mit - AWS KMS Schlüsseln (DSSE-KMS) verschlüsselt werden, verwenden die Replikatobjekte im Ziel-Bucket denselben Verschlüsselungstyp wie die Quellobjekte. Die Einstellungen der Standard-Verschlüsselung des Ziel-Buckets werden nicht verwendet.

Mit SSE-C verschlüsselte Objekte replizieren

Die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) gestattet Ihnen, eigene Verschlüsselungsschlüssel zu verwalten. Mit SSE-C verwalten Sie die Schlüssel, während Amazon S3 den Verschlüsselungs- und Entschlüsselungsprozess verwaltet. Sie müssen einen Verschlüsselungsschlüssel als Teil Ihrer Anfrage angeben, brauchen aber keinen Code zu schreiben, um die Objektverschlüsselung oder -entschlüsselung durchzuführen. Wenn Sie ein Objekt hochladen, verschlüsselt Amazon S3 das Objekt mithilfe des von Ihnen angegebenen Schlüssels. Amazon S3 löscht diesen Schlüssel dann aus dem Speicher. Wenn Sie ein Objekt abrufen, müssen Sie denselben Verschlüsselungsschlüssel als Teil Ihrer Anfrage angeben. Weitere Informationen finden Sie unter the section called “Vom Kunden bereitgestellte Verschlüsselungsschlüssel (SSE-C)”.

Die S3-Replikation unterstützt Objekte, die mit SSE-C verschlüsselt sind. Sie können die SSE-C-Objektreplikation in der Amazon S3-Konsole oder mit den - AWS SDKs auf die gleiche Weise konfigurieren, wie Sie die Replikation für unverschlüsselte Objekte konfigurieren. Es gibt keine zusätzlichen SSE-C-Berechtigungen, die über das hinausgehen, was derzeit für die Replikation erforderlich ist.

Die S3-Replikation repliziert automatisch neu hochgeladene mit SSE-C verschlüsselte Objekte, sofern sie wie in Ihrer S3-Replikationskonfiguration angegeben infrage kommen. Verwenden Sie die S3-Batchreplikation, um vorhandene Objekte in Ihren Buckets zu replizieren. Weitere Informationen zum Replizieren von Objekten finden Sie unter the section called “Einrichten der Replikation” und the section called “Replizieren vorhandener Objekte”.

Für die Replikation von SSE-C-Objekten fallen keine zusätzlichen Gebühren an. Einzelheiten zu den Replikationspreisen finden Sie auf der Seite Amazon S3 – Preise.

Replizieren von mit SSE-S3, SSE-KMS oder DSSE-KMS verschlüsselten Objekten

Standardmäßig repliziert Amazon S3 keine Objekte, die mit SSE-KMS oder DSSE-KMS verschlüsselt sind. In diesem Abschnitt werden die zusätzlichen Konfigurationselemente erörtert, die Sie hinzufügen können, um Amazon S3 anzuweisen, diese Objekte zu replizieren.

Ein Beispiel mit step-by-step Anweisungen finden Sie unter Replizieren verschlüsselter Objekte. Weitere Informationen zum Erstellen einer Replikationskonfiguration finden Sie unter Replizieren von Objekten.

Angeben zusätzlicher Informationen in der Replikations-Konfiguration

In der Replikations-Konfiguration machen Sie Folgendes:

- Fügen Sie im `Destination` Element in Ihrer Replikationskonfiguration die ID des symmetrischen, vom AWS KMS Kunden verwalteten Schlüssels hinzu, den Amazon S3 zum Verschlüsseln von Objektreplikaten verwenden soll, wie im folgenden Beispiel für eine Replikationskonfiguration gezeigt.
- Melden Sie sich ausdrücklich an, indem Sie die Replikation von Objekten aktivieren, die mit KMS-Schlüsseln (SSE-KMS oder DSSE-KMS) verschlüsselt wurden. Fügen Sie dazu das `SourceSelectionCriteria`-Element hinzu, wie im folgenden Beispiel einer Replikationskonfiguration dargestellt.

```
<ReplicationConfiguration>
   <Rule>
      ...
      <SourceSelectionCriteria>
         <SseKmsEncryptedObjects>
           <Status>Enabled</Status>
         </SseKmsEncryptedObjects>
      </SourceSelectionCriteria>

      <Destination>
          ...
          <EncryptionConfiguration>
             <ReplicaKmsKeyID>AWS KMS key ARN or Key Alias ARN that's in the same AWS-
Region as the destination bucket.</ReplicaKmsKeyID>
          </EncryptionConfiguration>
       </Destination>
      ...
```

```
   </Rule>
</ReplicationConfiguration>
```

> **Important**
>
> Der KMS-Schlüssel muss in derselben AWS-Region wie die Ziel-Buckets erstellt worden sein. Der KMS-Schlüssel muss gültig sein. Die `PutBucketReplication`-API-Operation überprüft nicht die Gültigkeit von KMS-Schlüsseln. Wenn Sie einen ungültigen KMS-Schlüssel verwenden, erhalten Sie als Antwort den HTTP-`200 OK`-Statuscode, aber die Replikation schlägt fehl.

Das folgende Beispiel zeigt eine Replikationskonfiguration, die optionale Konfigurationselemente enthält. Diese Replikations-Konfiguration weist eine Regel auf. Die Regel wird auf alle Objekte mit dem Schlüsselpräfix `Tax` angewendet. Amazon S3 verwendet die angegebene AWS KMS key -ID zum Verschlüsseln dieser Objektreplikate.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration>
   <Role>arn:aws:iam::account-id:role/role-name</Role>
   <Rule>
      <ID>Rule-1</ID>
      <Priority>1</Priority>
      <Status>Enabled</Status>
      <DeleteMarkerReplication>
         <Status>Disabled</Status>
      </DeleteMarkerReplication>
      <Filter>
         <Prefix>Tax</Prefix>
      </Filter>
      <Destination>
         <Bucket>arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET</Bucket>
         <EncryptionConfiguration>
            <ReplicaKmsKeyID>AWS KMS key ARN or Key Alias ARN that's in the same AWS-
Region as the destination bucket. (S3 uses this key to encrypt object replicas.)</
ReplicaKmsKeyID>
         </EncryptionConfiguration>
      </Destination>
      <SourceSelectionCriteria>
         <SseKmsEncryptedObjects>
            <Status>Enabled</Status>
```

```
          </SseKmsEncryptedObjects>
       </SourceSelectionCriteria>
    </Rule>
</ReplicationConfiguration>
```

Erteilen zusätzlicher Berechtigungen für die IAM-Rolle

Um Objekte zu replizieren, die im Ruhezustand mit SSE-S3, SSE-KMS oder DSSE-KMS verschlüsselt sind, erteilen Sie der AWS Identity and Access Management (IAM)-Rolle, die Sie in der Replikationskonfiguration angeben, die folgenden zusätzlichen Berechtigungen. Sie erteilen diese Berechtigungen, indem Sie die mit der IAM-Rolle verknüpfte Berechtigungsrichtlinie aktualisieren.

- **`s3:GetObjectVersionForReplication`**-Aktion für Quellobjekte – Diese Aktion ermöglicht Amazon S3, sowohl unverschlüsselte Objekte als auch Objekte, die mit serverseitiger Verschlüsselung mit SSE-S3, SSE-KMS oder DSSE-KMS erstellt wurden, zu replizieren.

  > **Note**
  >
  > Wir empfehlen, dass Sie die Aktion `s3:GetObjectVersionForReplication` statt der Aktion `s3:GetObjectVersion` verwenden, da `s3:GetObjectVersionForReplication` Amazon S3 nur die minimalen Berechtigungen bereitstellt, die für eine Replikation nötig sind. Darüber hinaus ermöglicht die Aktion `s3:GetObjectVersion` die Replikation von unverschlüsselten und über SSE-S3 verschlüsselten Objekten, nicht aber von Objekten, die mit KMS-Schlüsseln (SSE-KMS oder DSSE-KMS) verschlüsselt wurden.

- **`kms:Decrypt`** - und -**`kms:Encrypt`** AWS KMS Aktionen für die KMS-Schlüssel
  - Sie müssen `kms:Decrypt`-Berechtigungen für den AWS KMS key gewähren, der zum Entschlüsseln des Quellobjekts verwendet wird.
  - Sie müssen `kms:Encrypt`-Berechtigungen für den AWS KMS key gewähren, der zum Verschlüsseln des Objektreplikats verwendet wird.
- **`kms:GenerateDataKey`**-Aktion zur Replikation von Klartextobjekten – Wenn Sie Klartextobjekte in einen Bucket replizieren, für den die SSE-KMS- oder DSSE-KMS-Verschlüsselung standardmäßig aktiviert ist, müssen Sie die `kms:GenerateDataKey`-Berechtigung für den Zielverschlüsselungskontext und den KMS-Schlüssel zur IAM-Richtlinie hinzufügen.

Wir empfehlen, diese Berechtigungen mithilfe von AWS KMS Bedingungsschlüsseln nur auf die Ziel-Buckets und -Objekte zu beschränken. Das AWS-Konto , dem die IAM-Rolle gehört, muss über Berechtigungen für die `kms:Decrypt` Aktionen `kms:Encrypt` und für die KMS-Schlüssel verfügen, die in der Richtlinie aufgeführt sind. Wenn die KMS-Schlüssel einem anderen gehören AWS-Konto, muss der Besitzer der KMS-Schlüssel diese Berechtigungen dem gewähren AWS-Konto , dem die IAM-Rolle gehört. Weitere Informationen zum Verwalten des Zugriffs auf diese KMS-Schlüssel finden Sie unter Verwenden von IAM-Richtlinien mit AWS KMS im AWS Key Management Service - Entwicklerhandbuch.

S3 Bucket-Schlüssel und Replikation

Um die Replikation mit einem S3-Bucket-Schlüssel zu verwenden, muss die AWS KMS key Richtlinie für den KMS-Schlüssel, der zum Verschlüsseln des Objektreplikats verwendet wird, die -`kms:Decrypt`Berechtigung für den aufrufenden Prinzipal enthalten. Der Aufruf zur `kms:Decrypt` Überprüfung der Integrität des S3-Bucket-Schlüssels, bevor er verwendet wird. Weitere Informationen finden Sie unter Verwenden eines S3-Bucket-Schlüssels mit Replikation.

Wenn ein S3-Bucket-Schlüssel für den Quell- oder Ziel-Bucket aktiviert ist, ist der Verschlüsselungskontext der Amazon-Ressourcenname (ARN) des Buckets, nicht der Objekt-ARN (z. B. `arn:aws:s3:::`*`bucket_ARN`*). Sie müssen Ihre IAM-Richtlinien aktualisieren, um den Bucket-ARN für den Verschlüsselungskontext verwenden zu können:

```
"kms:EncryptionContext:aws:s3:arn": [
"arn:aws:s3:::bucket_ARN"
]
```

Weitere Informationen finden Sie unter Verschlüsselungskontext (x-amz-server-side-encryption-context) (im Abschnitt „Verwenden der REST-API“ und unter Änderungen, die Sie vor dem Aktivieren eines S3-Bucket-Schlüssels beachten sollten.

Beispielrichtlinien – Verwenden von SSE-S3 und SSE-KMS bei der Replikation

Die folgenden IAM-Beispielrichtlinien zeigen Anweisungen für die Verwendung von SSE-S3 und SSE-KMS bei der Replikation.

Example – Verwenden von SSE-KMS mit separaten Ziel-Buckets

Die folgende Beispielrichtlinie zeigt Anweisungen zur Verwendung von SSE-KMS mit separaten Ziel-Buckets.

```
{
```

```
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action": ["kms:Decrypt"],
         "Effect": "Allow",
         "Condition": {
             "StringLike": {
                  "kms:ViaService": "s3.source-bucket-region.amazonaws.com",
                  "kms:EncryptionContext:aws:s3:arn": [
                      "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET/key-prefix1*"
                  ]
             }
          },
          "Resource": [
             "List of AWS KMS key ARNs that are used to encrypt source objects."
          ]
      },
      {
         "Action": ["kms:Encrypt"],
         "Effect": "Allow",
         "Condition": {
            "StringLike": {
               "kms:ViaService": "s3.destination-bucket-1-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn": [
                  "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET1/key-prefix1*"
               ]
         }
         },
         "Resource": [
            "AWS KMS key ARNs (in the same AWS-Region as destination bucket 1). Used to
encrypt object replicas created in destination bucket 1."
         ]
      },
      {
         "Action": ["kms:Encrypt"],
         "Effect": "Allow",
         "Condition": {
            "StringLike": {
               "kms:ViaService": "s3.destination-bucket-2-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn": [
                  "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET2/key-prefix1*"
               ]
         }
         },
```

```
            "Resource": [
               "AWS KMS key ARNs (in the same AWS-Region as destination bucket 2). Used to
 encrypt object replicas created in destination bucket 2."
            ]
         }
      ]
}
```

Example – Replizieren von mit SSE-S3 und SSE-KMS erstellten Objekten

Im Folgenden sehen Sie eine vollständige IAM-Richtlinie, die die erforderlichen Berechtigungen zum Replizieren von unverschlüsselten Objekten, Objekten, die mit SSE-S3 erstellt wurden, und Objekten, die mit SSE-KMS erstellt wurden, gewährt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET/key-prefix1*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete"
         ],
```

```
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/key-prefix1*"
      },
      {
         "Action":[
            "kms:Decrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.source-bucket-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET/key-prefix1*"
               ]
            }
         },
         "Resource":[
           "List of the AWS KMS key ARNs that are used to encrypt source objects."
         ]
      },
      {
         "Action":[
            "kms:Encrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.destination-bucket-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/prefix1*"
               ]
            }
         },
         "Resource":[
            "AWS KMS key ARNs (in the same AWS-Region as the destination bucket) to use
 for encrypting object replicas"
         ]
      }
   ]
}
```

Example – Replizieren von Objekten mit S3-Bucket-Schlüsseln

Im Folgenden sehen Sie eine vollständige IAM-Richtlinie, die die erforderlichen Berechtigungen zum Replizieren von Objekten mit S3-Bucket-Schlüsseln gewährt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET/key-prefix1*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/key-prefix1*"
      },
      {
         "Action":[
            "kms:Decrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.source-bucket-region.amazonaws.com",
```

```
                "kms:EncryptionContext:aws:s3:arn":[
                   "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
                ]
             }
          },
          "Resource":[
            "List of the AWS KMS key ARNs that are used to encrypt source objects."
          ]
       },
       {
          "Action":[
             "kms:Encrypt"
          ],
          "Effect":"Allow",
          "Condition":{
             "StringLike":{
                "kms:ViaService":"s3.destination-bucket-region.amazonaws.com",
                "kms:EncryptionContext:aws:s3:arn":[
                   "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET"
                ]
             }
          },
          "Resource":[
             "AWS KMS key ARNs (in the same AWS-Region as the destination bucket) to use
 for encrypting object replicas"
          ]
       }
    ]
}
```

Erteilen von zusätzlichen Berechtigungen für kontenübergreifende Szenarien

In einem kontoübergreifenden Szenario, in dem die Quell- und Ziel-Buckets verschiedenen gehören AWS-Konten, können Sie einen KMS-Schlüssel verwenden, um Objektreplikate zu verschlüsseln. Der KMS-Schlüssel-Besitzer muss dem Besitzer des Quell-Buckets jedoch die Berechtigung erteilen, den KMS-Schlüssel zu verwenden.

> **Note**
>
> Über Von AWS verwaltete Schlüssel verschlüsselte Objekte können nicht kontoübergreifend geteilt werden, da Sie die Schlüsselrichtlinien nicht ändern können. Wenn Sie SSE-KMS-

Daten kontoübergreifend replizieren müssen, müssen Sie einen vom Kunden verwalteten Schlüssel von verwenden AWS KMS.

So erteilen Sie dem Quell-Bucket-Eigentümer die Berechtigung zur Verwendung des KMS-Schlüssels (AWS KMS -Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - AWS KMS Konsole unter https://console.aws.amazon.com/kms.
2. Um die zu ändern AWS-Region, verwenden Sie die Regionsauswahl in der oberen rechten Ecke der Seite.
3. Zum Anzeigen der Schlüssel in Ihrem Konto, die Sie erstellen und verwalten, wählen Sie im Navigationsbereich Customer managed keys (Vom Kunden verwaltete Schlüssel) aus.
4. Wählen Sie den KMS-Schlüssel aus.
5. Wählen Sie unter Allgemeine Konfiguration den Tab Schlüsselrichtlinie aus.
6. Scrollen Sie nach unten zu Andere AWS-Konten.
7. Wählen Sie Andere hinzufügen aus AWS-Konten.

   Das Dialogfeld Andere AWS-Konten wird angezeigt.
8. Wählen Sie im Dialogfeld Weitere hinzufügen aus AWS-Konto. Geben Sie für arn:aws:iam:: die Konto-ID des Quell-Buckets ein.
9. Wählen Sie Änderungen speichern aus.

So erteilen Sie dem Quell-Bucket-Eigentümer die Berechtigung zur Verwendung des KMS-Schlüssels (AWS CLI)

- Weitere Informationen zum Befehl `put-key-policy` AWS Command Line Interface (AWS CLI) finden Sie unter put-key-policy in der AWS CLI -Befehlsreferenz. Weitere Informationen über die zugrundeliegende `PutKeyPolicy`-API-Operation finden Sie unter PutKeyPolicy in der AWS Key Management Service -API-Referenz.

AWS KMS Überlegungen zu Transaktionskontingenten

Wenn Sie nach der Aktivierung der regionsübergreifenden Replikation (CRR) viele neue Objekte mit AWS KMS Verschlüsselung hinzufügen, kann es zu einer Drosselung kommen (HTTP-`503 Service Unavailable`Fehler). Die Drosselung erfolgt, wenn die Anzahl an AWS KMS -Transaktionen pro

Sekunde das aktuelle Kontingent überschreitet. Weitere Informationen finden Sie unter Kontingente im AWS Key Management Service -Entwicklerhandbuch.

Zum Anfordern einer Erhöhung für ein Kontingent verwenden Sie Service-Quotas verwenden. Weitere Informationen finden Sie unter Anfordern einer Kontingenterhöhung. Wenn Service Quotas in Ihrer Region nicht unterstützt wird, öffnen Sie einen - AWS Support Fall.

Abrufen von Replikationsstatusinformationen

Der Replikationsstatus hilft Ihnen, den aktuellen Status eines derzeit replizierten Objekts zu bestimmen. Der Replikationsstatus eines Quellobjekts gibt entweder `PENDING`, `COMPLETED` oder `FAILED` zurück. Der Replikationsstatus eines Replikats gibt `REPLICA` zurück.

Themen

- Übersicht über den Replikationsstatus
- Replikationsstatus bei einer Replikation auf mehrere Ziel-Buckets
- Replikationsstatus, wenn die Synchronisierung von Amazon-S3-Replikatänderungen aktiviert ist
- Finden des Replikationsstatus

Übersicht über den Replikationsstatus

Bei der Replikation haben Sie einen Quell-Bucket, auf dem Sie die Replikation und das Ziel konfigurieren, an dem Amazon S3 Objekte repliziert. Wenn Sie ein Objekt (mit `GET` Objekt) oder Objektmetadaten (mit `HEAD` Objekt) von diesen Buckets anfordern, gibt Amazon S3 den Header `x-amz-replication-status` wie folgt in der Antwort zurück:

- Wenn Sie ein Objekt aus dem Quell-Bucket anfordern, gibt Amazon S3 den Header `x-amz-replication-status` zurück, wenn das Objekt in der Anforderung für die Replikation geeignet ist.

  Nehmen wir beispielsweise an, dass Sie in Ihrer Replikationskonfiguration das Objektpräfix `TaxDocs` angeben, um Amazon S3 anzuweisen, nur Objekte mit dem Schlüsselnamenpräfix `TaxDocs` zu replizieren. Alle Objekte mit diesem Schlüsselnamenpräfix, die Sie hochladen, z. B. `TaxDocs/document1.pdf`, werden repliziert. Für Objektanforderungen mit diesem Schlüsselnamenpräfix gibt Amazon S3 den Header `x-amz-replication-status` mit einem der folgenden Werte für den Replikationsstatus des Objekts zurück: `PENDING`, `COMPLETED` oder `FAILED`.

> **Note**
>
> Wenn nach dem Hochladen eines Objekts die Objektreplikation fehlschlägt, können Sie die fehlgeschlagene Replikation nicht erneut durchzuführen versuchen. Sie müssen das Objekt erneut hochladen. Bei Problemen wie fehlenden Replikationsrollen-Berechtigungen, AWS KMS -Berechtigungen oder Bucket-Berechtigungen gehen Objekte in den Status `FAILED` über. Bei temporären Fehlern, z. B. wenn ein Bucket oder eine Region nicht verfügbar ist, geht der Replikationsstatus nicht in `FAILED` über, sondern verbleibt bei `PENDING`. Wenn die Ressource wieder online ist, setzt S3 die Replikation dieser Objekte fort.

- Wenn Sie ein Objekt aus einem Ziel-Bucket anfordern und es sich bei dem Objekt Ihrer Anforderung um ein Replikat handelt, das Amazon S3 erstellt hat, gibt Amazon S3 den Header `x-amz-replication-status` mit dem Wert `REPLICA` zurück.

> **Note**
>
> Bevor Sie ein Objekt aus einem Quell-Bucket löschen, bei dem die Replikation aktiviert ist, sollten Sie den Replikationsstatus des Objekts überprüfen, um sicherzustellen, dass das Objekt repliziert wurde.
> Wenn die Lebenszykluskonfiguration auf dem Quell-Bucket aktiviert ist, setzt Amazon S3 alle Lebenszyklusaktionen aus, bis der Status des Objekts als `COMPLETED` oder `FAILED` gekennzeichnet wird.

Replikationsstatus bei einer Replikation auf mehrere Ziel-Buckets

Wenn Sie Objekte in mehrere Ziel-Buckets replizieren, verhält sich der Header `x-amz-replication-status` anders. Der Header des Quellobjekts gibt den Wert `COMPLETED` nur zurück, wenn die Replikation in alle Ziele erfolgreich ist. Der Header bleibt auf dem Wert `PENDING`, bis die Replikation für alle Ziele abgeschlossen ist. Wenn die Replikation bei einem oder mehreren Zielen fehlschlägt, liefert der Header `FAILED`.

Replikationsstatus, wenn die Synchronisierung von Amazon-S3-Replikatänderungen aktiviert ist

Wenn in Ihren Replikationsregeln die Synchronisierung von Amazon-S3-Replikatänderungen aktiviert ist, können Replikate einen anderen Status als `REPLICA` melden. Wenn Änderungen an

Metadaten gerade repliziert werden, gibt der `x-amz-replication-status`-Header den Wert `PENDING` zurück. Wenn die Synchronisierung der Replikatänderung Metadaten nicht repliziert, gibt der Header `FAILED` zurück. Wenn Metadaten korrekt repliziert werden, geben die Replikate den Header `REPLICA` zurück.

Finden des Replikationsstatus

Verwenden Sie das Amazon-S3-Inventory-Tool, um den Replikationsstatus der Objekte in einem Bucket abzurufen. Amazon S3 sendet eine CSV-Datei an den Ziel-Bucket, den Sie in der Bestands-Konfiguration angeben. Sie können auch Amazon Athena verwenden, um den Replikationsstatus im Bestandsbericht abzufragen. Weitere Informationen zu Amazon S3 Inventory finden Sie unter Amazon S3 Inventory.

Sie können den Replikationsstatus des Objekts auch mithilfe der Konsole, der AWS Command Line Interface (AWS CLI) oder des AWS SDK ermitteln.

Verwenden der S3-Konsole

In der S3-Konsole können Sie den Replikationsstatus für ein Objekt auf der Objektseite Details unter Objektverwaltungsübersichtanzeigen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus.
3. Wählen Sie in der Liste Objekte den Objektnamen aus.
4. Auf der Registerkarte Properties (Eigenschaften) unter Object management overview (Objektverwaltungsübersicht) sehen Sie den Replication status (Replikationsstatus).

Verwenden der AWS CLI

Verwenden Sie den Befehl `head-object`, um Objektmetadaten abzurufen.

```
aws s3api head-object --bucket source-bucket --key object-key --version-id object-version-id
```

Der Befehl gibt Informationen zu Objektmetadaten, einschließlich des `ReplicationStatus`, zurück, wie in der folgenden Beispielantwort dargestellt:

```
{
   "AcceptRanges":"bytes",
   "ContentType":"image/jpeg",
   "LastModified":"Mon, 23 Mar 2015 21:02:29 GMT",
   "ContentLength":3191,
   "ReplicationStatus":"COMPLETED",
   "VersionId":"jfnW.HIMOfYiD_9rGbSkmroXsFj3fqZ.",
   "ETag":"\"6805f2cfc46c0f04559748bb039d69ae\"",
   "Metadata":{

   }
}
```

Verwenden der AWS SDKs

Die folgenden Codefragmente erhalten den Replikationsstatus mit bzw. AWS SDK for Java AWS SDK for .NET.

Java

```
GetObjectMetadataRequest metadataRequest = new GetObjectMetadataRequest(bucketName,
 key);
ObjectMetadata metadata = s3Client.getObjectMetadata(metadataRequest);

System.out.println("Replication Status : " +
 metadata.getRawMetadataValue(Headers.OBJECT_REPLICATION_STATUS));
```

.NET

```
GetObjectMetadataRequest getmetadataRequest = new GetObjectMetadataRequest
    {
         BucketName = sourceBucket,
         Key        = objectKey
    };

GetObjectMetadataResponse getmetadataResponse =
 client.GetObjectMetadata(getmetadataRequest);
Console.WriteLine("Object replication status: {0}",
 getmetadataResponse.ReplicationStatus);
```

Weitere Überlegungen

Amazon S3 unterstützt auch Bucket-Konfigurationen für Folgendes:

- Versioning – Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.
- Website-Hosting – Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.
- Bucket-Zugriff über eine Bucket-Richtlinie oder Zugriffskontrollliste (ACL) – Weitere Informationen finden Sie unter Bucket-Richtlinien und Benutzerrichtlinien und Zugriffskontrolllisten (ACL) – Übersicht.
- Protokollspeicher – Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.
- Lebenszyklusverwaltung für Objekte in einem Bucket – Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

In diesem Thema wird erläutert, wie die Bucket-Replikations-Konfiguration das Verhalten dieser Bucket-Konfigurationen beeinflusst.

Themen

- Lebenszyklus-Konfiguration und Objektreplikate
- Versioning-Konfiguration und Replikations-Konfiguration
- Verwenden der S3-Replikation mit S3-Intelligent-Tiering
- Protokollierungskonfiguration und Replikations-Konfiguration
- CRR und die Zielregion
- Pausieren einer Replikation

Lebenszyklus-Konfiguration und Objektreplikate

Die von Amazon S3 für die Replikation eines Objekts benötigte Zeit hängt von der Größe des Objekts ab. Bei großen Objekten kann dies mehrere Stunden dauern. Auch wenn es möglicherweise etwas dauert, bis ein Replikat im Ziel verfügbar ist, entspricht die Erstellungszeit des Replikats der Erstellungszeit des entsprechenden Objekts im Quell-Bucket. Wenn eine Lebenszyklus-Konfiguration für einen Ziel-Bucket aktiviert ist, berücksichtigen die Lebenszyklusregeln die ursprüngliche Erstellungszeit des Objekts, nicht den Zeitpunkt, zu dem das Replikat im Ziel-Bucket verfügbar wurde.

Die Replikations-Konfiguration erfordert einen Bucket mit aktiviertem Versioning. Berücksichtigen Sie bei der Aktivierung des Versioning in einem Bucket Folgendes:

- Wenn eine Lebenszyklus-Konfiguration für den Ablauf von Objekten vorhanden ist, nachdem Sie die Versionsverwaltung aktiviert haben, fügen Sie eine `NonCurrentVersionExpiration`-Richtlinie hinzu, damit dasselbe Verhalten hinsichtlich einer dauerhaften Löschung beibehalten wird wie vor der Aktivierung der Versionsverwaltung.
- Wenn eine Übergangs-Lebenszyklus-Konfiguration vorhanden ist, nachdem Sie die Versionsverwaltung aktiviert haben, sollten Sie in Betracht ziehen, eine `NonCurrentVersionTransition`-Richtlinie hinzuzufügen.

Versioning-Konfiguration und Replikations-Konfiguration

Wenn Sie die Replikation auf einem Bucket konfigurieren, muss sowohl für den Quell- als auch für den Ziel-Bucket das Versioning aktiviert sein. Nachdem Sie sowohl im Quell- als auch im Ziel-Bucket das Versioning aktiviert und die Replikation im Quell-Bucket konfiguriert haben, werden Sie folgende Schwierigkeiten beobachten:

- Wenn Sie versuchen, das Versioning auf dem Quell-Bucket zu deaktivieren, gibt Amazon S3 einen Fehler zurück. Sie müssen die Replikations-Konfiguration entfernen, damit Sie das Versioning auf dem Quell-Bucket deaktivieren können.
- Wenn Sie das Versioning auf dem Ziel-Bucket deaktivieren, schlägt die Replikation fehl. Das Quellobjekt weist den Replikationsstatus `FAILED` auf.

Verwenden der S3-Replikation mit S3-Intelligent-Tiering

S3-Intelligent-Tiering ist eine Speicherklasse, die entworfen wurde, um die Speicherkosten zu optimieren, indem sie Daten automatisch zur kostengünstigsten Zugriffsstufe verschiebt. Gegen eine geringe monatliche Gebühr für Objektüberwachung und Automatisierung überwacht S3 Intelligent-Tiering die Zugriffsmuster und verschiebt die Objekte, auf die nicht zugegriffen wurde, automatisch in kostengünstigere Zugriffsebenen.

Das Replizieren von in S3-Intelligent-Tiering gespeicherten Objekte durch eine S3-Batch-Replikation oder das Aufrufen von CopyObject oder UploadPartCopy stellt Zugriff dar. In diesen Fällen werden die Quellobjekte der Kopier- oder Replikationsvorgänge hochgestuft.

Weitere Informationen zu S3-Intelligent-Tiering finden Sie unter Amazon S3 Intelligent Tiering.

Protokollierungskonfiguration und Replikations-Konfiguration

Wenn Amazon S3 Protokolle an einen Bucket übermittelt, für den die Replikation aktiviert ist, repliziert der Service die Protokollobjekte.

Wenn Sie Server-Zugriffsprotokolle (Protokollieren von Anfragen mit Server-Zugriffsprotokollierung) oder AWS CloudTrail -Protokolle (Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail) im Quell- oder Ziel-Bucket aktiviert haben, schließt Amazon S3 die replikationsbezogenen Anforderungen in die Protokolle mit ein. Beispiel: Amazon S3 protokolliert jedes Objekt, das es repliziert.

CRR und die Zielregion

Die regionsübergreifende Replikation (CRR) von Amazon S3 wird verwendet, um Objekte über S3-Buckets in verschiedenen zu kopieren AWS-Regionen. Sie können die Region für Ihren Ziel-Bucket entweder auf der Basis Ihrer geschäftlichen Anforderungen oder von Kostenaspekten auswählen. Beispielsweise variieren die Gebühren für eine regionsübergreifende Datenübertragung je nach gewählten Regionen.

Angenommen, Sie wählen USA Ost (Nord-Virginia) (us-east-1) als Region für Ihren Quell-Bucket aus. Wenn Sie USA West (Oregon) (us-west-2) als Region für Ihre Ziel-Buckets auswählen, zahlen Sie mehr, als wenn Sie die Region USA Ost (Ohio) (us-east-2) auswählen. Weitere Informationen zu Preisen finden Sie im Abschnitt „Datenübertragungspreise“ unter Amazon S3 – Preise.

Für eine Replikation innerhalb derselben Region (SRR) fallen keine Datenübertragungskosten an.

Pausieren einer Replikation

Um mit der Replikation vorübergehend zu pausieren, deaktivieren Sie die entsprechende Regel in der Replikations-Konfiguration.

Wenn die Replikation aktiviert ist und Sie die IAM-Rolle entfernen, die Amazon S3 die erforderlichen Berechtigungen gewährt, schlägt die Replikation fehl. Amazon S3 meldet den Replikationsstatus für betroffene Objekte als `FAILED`.

Kategorisieren des Speichers mithilfe von Markierungen

Markieren Sie Objekte, um Speicher zu kategorisieren. Jeder Tag ist ein Schlüssel/Wert-Paar.

Sie können neuen Objekten Markierungen hinzufügen, wenn Sie sie hochladen, Sie können Markierungen aber auch vorhandenen Objekten hinzufügen.

- Sie können einem Objekt bis zu 10 Markierungen zuordnen. Einem Objekt zugeordnete Markierungen müssen eindeutige Tag-Schlüssel haben.
- Ein Tag-Schlüssel kann maximal 128 Unicode-Zeichen lang sein, und die Tag-Werte können bis zu 256 Unicode-Zeichen lang sein. Amazon-S3-Objekt-Tags werden intern in UTF-16 dargestellt. Beachten Sie, dass Zeichen in UTF-16 entweder 1 oder 2 Zeichenpositionen einnehmen.
- Bei Schlüsseln und Werten wird die Groß-/Kleinschreibung berücksichtigt.
- Weitere Informationen zu Tag-Einschränkungen finden Sie unter User-Defined Tag Restrictions (Einschränkungen benutzerdefinierter Markierungen).

Beispiele

Betrachten Sie die folgenden Beispiele für die Markierung:

Example PHI-Informationen

Angenommen, ein Objekt enthält PHI-Daten (Protected Health Information, geschützte Gesundheitsdaten). Sie könnten das Objekt unter Verwendung des folgenden-Schlüssel-Wert-Paares markieren:

```
PHI=True
```

oder

```
Classification=PHI
```

Example Projektdateien

Angenommen, Sie speichern Projektdateien in Ihrem S3-Bucket. Sie könnten diese Objekte mit einem Schlüssel namens `Project` und einem Wert markieren, wie nachfolgend gezeigt:

```
Project=Blue
```

Example Mehrere Markierungen

Sie können einem Objekt mehrere Markierungen hinzufügen, wie nachfolgend gezeigt:

```
Project=x
Classification=confidential
```

Schlüsselnamen-Präfixe und -Markierungen

Mit Schlüsselnamenpräfixe können Sie auch Speicher kategorisieren. Allerdings sind Präfix-basierte Kategorisierungen eindimensional. Sehen Sie sich die folgenden Objektschlüsselnamen an:

```
photos/photo1.jpg
project/projectx/document.pdf
project/projecty/document2.pdf
```

Dieses Schlüsselnamen haben die Präfixe `photos/`, `project/projectx/` und `project/projecty/`. Diese Präfixe unterstützen eine eindimensionale Kategorisierung. Das bedeutet, alles unter einem Präfix ist eine Kategorie. Beispielsweise identifiziert das Präfix `project/projectx` alle Dokumente, die zu Projekt X gehören.

Mit der Markierung erhalten Sie jetzt eine weitere Dimension. Wenn Sie photo1 in der Kategorie project x anlegen wollen, können Sie das Objekt entsprechend markieren.

Zusätzliche Vorteile

Neben der Datenklassifizierung bietet die Markierung auch noch weitere Vorteile.

- Objekt-Markierungen bieten eine differenzierte Zugriffskontrolle für Berechtigungen. Sie könnten z. B. einem Benutzer Berechtigungen erteilen, nur Objekte mit bestimmten Markierungen zu lesen.
- Objekt-Markierungen unterstützen ein differenziertes Objektlebenszyklusmanagement, bei dem Sie in einer Lebenszyklusregel zusätzlich zum Schlüsselnamenpräfix einen auf Markierungen basierenden Filter angeben können.
- Wenn Sie Amazon-S3-Analysen verwenden, können Sie Filter konfigurieren, um Objekte für die Analyse nach Objekt-Markierungen, nach Schlüsselnamen-Präfix oder nach Präfix und Markierungen zu gruppieren.
- Sie können auch Amazon- CloudWatch Metriken anpassen, um Informationen nach bestimmten Tag-Filtern anzuzeigen. Die folgenden Abschnitte stellen Details bereit.

> **Important**
>
> Es ist akzeptabel, Markierungen zu verwenden, um Objekte mit vertraulichen Daten zu markieren (z. B. personenbezogene Informationen (PII) oder geschützte Gesundheitsinformationen (PHI)). Diese Markierungen sollten jedoch selbst keine vertraulichen Daten enthalten.

Hinzufügen von Objekt-Tag-Sätzen zu mehreren Amazon-S3-Objekten mit einer einzigen Anfrage

Zum Hinzufügen von Objekt-Tag-Mengen zu mehr als einem Amazon-S3-Objekt mit einer einzelnen Anforderung können Sie S3-Batchoperationen verwenden. Sie stellen S3 Batch Operations eine Liste von Objekten zur Verfügung, für die Vorgänge ausgeführt werden sollen. S3-Batchoperationen rufen die entsprechende API-Operation auf, um die angegebene Operation auszuführen. Ein einzelner Batch-Vorgangsauftrag kann die angegebene Operation für Milliarden von Objekten ausführen, die Exabytes von Daten enthalten.

Die Funktion „S3-Batchoperationen“ verfolgt den Fortschritt, versendet Benachrichtigungen und speichert einen detaillierten Abschlussbericht zu allen Aktionen. Sie profitieren von einer vollständig verwalteten, prüfbaren und serverlosen Umgebung. Sie können S3-Batchoperationen über die Amazon S3-Konsole, AWS CLI, die AWS-SDKs oder REST-API verwenden. Weitere Informationen finden Sie unter the section called “Grundlagen von BatchVorgänge”.

Weitere Informationen über Objekt-Markierungen finden Sie unter Verwalten von Objekt-Markierungen.

API-Operationen für die Objektmarkierung

Amazon S3 unterstützt die folgenden API-Operationen, die spezifisch für das Objekt-Tagging sind:

Objekt-API-Operationen

- PUT Object tagging – Ersetzt Markierungen auf einem Objekt. Sie geben die Markierungen im Anfragerumpf an. Es gibt zwei unterschiedliche Szenarien der Objekt-Tag-Verwaltung unter Verwendung dieser API.
  - Objekt hat keine Markierungen – Mit Hilfe dieser API können Sie einem Objekt verschiedene Markierungen hinzufügen (das Objekt hat keine vorherigen Markierungen).
  - Das Objekt hat eine Menge vorhandener Markierungen – Um die vorhandene Tag-Menge zu ändern, müssen Sie zuerst die vorhandene Tag-Menge abrufen, sie auf der Client-Seite ändern und diese API dann verwenden, um die Tag-Menge zu ersetzen.

    > **Note**
    >
    > Wenn Sie diese Anforderung mit einer leeren Tag-Menge senden, löscht Amazon S3 die vorhandene Tag-Menge für das Objekt. Wenn Sie diese Methode verwenden, wird eine Tier 1-Anforderung (PUT) berechnet. Weitere Informationen finden Sie unter Amazon S3 – Preise.

> Die Anforderung DELETE Object tagging wird empfohlen, weil sie das gleiche Ergebnis liefert, aber keine Kosten verursacht.

- GET Object tagging – Gibt die einem Objekt zugeordnete Tag-Menge zurück. Amazon S3 gibt die Objekt-Markierungen im Antworttext zurück.
- DELETE Object tagging – Löscht die mit einem Tag verbundenen Tag-Menge.

Andere API-Operationen, die Markieren unterstützen

- PUT Object und Initiate Multipart Upload– Sie können beim Erstellen von Objekten Markierungen angeben. Sie geben Markierungen unter Verwendung des Anfrage-Headers `x-amz-tagging` an.
- GET Object – Statt die Tag-Menge zurückzugeben, gibt Amazon S3 den Objekt-Tag-Zähler im `x-amz-tag-count`-Header zurück (nur dann, wenn der Auftraggeber die Berechtigung hat, Markierungen zu lesen), weil die Header-Antwortgröße auf 8 KB begrenzt ist. Wenn Sie die Markierungen anzeigen möchten, erstellen Sie eine weitere Anfrage für die API-Operation GET Object tagging.
- POST Object – Sie können Markierungen in Ihrer POST-Anfrage angeben.

  So lange die von Ihnen geforderten Markierungen die 8-KB-Größenbeschränkung der HTTP-Anfrageheader nicht überschreitet, können Sie die `PUT Object` -API verwenden, um Objekte mit Markierungen zu erstellen. Wenn die von Ihnen angegebenen Markierungen die Größenbeschränkung des Headers überschreiten, können Sie diese POST-Methode verwenden, wobei Sie die Markierungen in den Rumpf aufnehmen.

  PUT Object - Copy – Sie können die `x-amz-tagging-directive` in Ihre Anfrage aufnehmen, um Amazon S3 anzuweisen, die Markierungen zu kopieren (Standardverhalten) oder durch eine neue, in der Anfrage angegebene Tag-Menge zu ersetzen.

Beachten Sie Folgendes:

- Die S3-Objektmarkierung ist hochgradig konsistent. Weitere Informationen finden Sie unter Amazon S3-Datenkonsistenzmodell.

Zusätzliche Konfigurationen

Dieser Abschnitt erklärt, was die Objektmarkierung für andere Konfigurationen bedeutet.

Objektmarkierung und Lebenszyklusverwaltung

In der Bucket-Lebenszyklus-Konfiguration können Sie einen Filter angeben, um eine Untermenge von Objekten auszuwählen, auf die die Regel anzuwenden ist. Sie können einen Filter basierend auf den Schlüsselnamenpräfixen, Objekt-Markierungen oder beidem angeben.

Angenommen, Sie speichern Fotos (Rohdaten und im fertigen Format) in Ihrem Amazon-S3-Bucket. Sie könnten diese Objekte wie folgt markieren:

```
phototype=raw
or
phototype=finished
```

Sie könnten festlegen, dass die Rohdaten der Fotos irgendwann nach der Erstellung in S3 Glacier archiviert werden. Sie können eine Lebenszyklusregel mit einem Filter konfigurieren, der die Untermenge der Objekte mit dem Schlüsselnamenpräfix (`photos/`) identifiziert, die ein spezifisches Tag (`phototype=raw`) haben.

Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Objektmarkierung und -replikation

Wenn Sie auf Ihrem Bucket die Replikation konfiguriert haben, repliziert Amazon S3 Markierungen, sofern Sie Amazon S3 die Berechtigung zum Lesen der Markierungen erteilt haben. Weitere Informationen finden Sie unter Einrichten der Replikation.

Ereignisbenachrichtigungen einer Objektmarkierung

Sie können eine Amazon-S3-Ereignisbenachrichtigung einrichten, um benachrichtigt zu werden, wenn eine Objektmarkierung hinzugefügt oder aus einem Objekt gelöscht wird. Der `s3:ObjectTagging:Put`-Ereignistyp benachrichtigt Sie, wenn ein Tag auf einem Objekt PUTiert wird oder wenn ein vorhandener Tag aktualisiert wird. Der `s3:ObjectTagging:Delete`-Ereignistyp benachrichtigt Sie, wenn ein Tag aus einem Objekt entfernt wird. Weitere Informationen finden Sie unter Aktivieren von Ereignisbenachrichtigungen.

Weitere Informationen über die Objektmarkierung finden Sie in den folgenden Themen:

Themen

- Markierungs- und Zugriffskontrollrichtlinien
- Verwalten von Objekt-Markierungen

Markierungs- und Zugriffskontrollrichtlinien

Sie können außerdem Berechtigungsrichtlinien (Bucket- und Benutzerrichtlinien) verwenden, um Berechtigungen für das Objekt-Tagging zu verwalten. Informationen über Richtlinienaktionen finden Sie in den folgenden Themen:

- Objektoperationen
- Bucket-Operationen

Objekt-Markierungen bieten eine differenzierte Zugriffskontrolle für die Verwaltung von Berechtigungen. Sie können bedingte Berechtigungen basierend auf Objekt-Markierungen erteilen. Amazon S3 unterstützt die folgenden Bedingungsschlüssel, die Sie verwenden können, um bedingte Berechtigungen basierend auf Objekt-Markierungen zu erteilen.

- `s3:ExistingObjectTag/<tag-key>` – Verwenden Sie diesen Bedingungsschlüssel, um zu überprüfen, ob ein vorhandenes Objekt-Tag den spezifischen Tag-Schlüssel und -Wert besitzt.

  > **Note**
  >
  > Wenn Sie Berechtigungen für die `PUT Object`- und `DELETE Object`-Operationen erteilen, wird dieser Bedingungsschlüssel nicht unterstützt. Dies bedeutet, dass Sie keine Richtlinie erstellen können, um einem Benutzer zu gestatten, ein Objekt basierend auf seinen vorhandenen Markierungen zu löschen oder zu überschreiben.

- `s3:RequestObjectTagKeys` – Verwenden Sie diesen Bedingungsschlüssel, um die Tag-Schlüssel einzuschränken, die Sie für Objekte zulassen wollen. Dies ist nützlich, wenn Sie Objekten Tags mit der PutObjectTagging und PutObjectder und POST-Objektanforderungen hinzufügen.
- `s3:RequestObjectTag/<tag-key>` – Verwenden Sie diesen Bedingungsschlüssel, um die Tag-Schlüssel und -Werte einzuschränken, die Sie für Objekte zulassen wollen. Dies ist nützlich, wenn Sie Objekten Tags mit der PutObjectTagging und der PutObjectund POST-Bucket-Anforderungen hinzufügen.

Eine vollständige Liste der für den Amazon-S3-Service spezifischen Bedingungsschlüssel finden Sie unter Beispiele für Amazon-S3-Bedingungsschlüssel. Die folgenden Berechtigungsrichtlinien zeigen, wie die Objektmarkierung eine differenzierte Zugriffsberechtigungsverwaltung ermöglicht.

Example 1: Einem Benutzer nur das Lesen von Objekten gestatten, die einen bestimmten Tag-Schlüssel und -Wert besitzen

Die folgende Berechtigungsrichtlinie beschränkt einen Benutzer darauf, nur Objekte zu lesen, die den Tag-Schlüssel und -Wert `environment: production` haben. Diese Richtlinie verwendet den `s3:ExistingObjectTag`-Bedingungsschlüssel, um den Tag-Schlüssel und -Wert anzugeben.

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Principal": {
      "AWS": [
        "arn:aws:iam::111122223333:role/JohnDoe"
      ]
    },
    "Effect": "Allow",
    "Action": ["s3:GetObject", "s3:GetObjectVersion"],
    "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
    "Condition": {
      "StringEquals":
        {"s3:ExistingObjectTag/environment": "production"}
    }
  }
  ]
}
```

Example 2: Einschränken, welche Objekt-Tag-Schlüssel Benutzer hinzufügen können

Die folgende Berechtigungsrichtlinie erteilt einem Benutzer die Berechtigungen, die `s3:PutObjectTagging`-Aktion auszuführen, die dem Benutzer gestattet, einem vorhandenen Objekt Markierungen hinzuzufügen. Die Bedingung verwendet den Bedingungsschlüssel `s3:RequestObjectTagKeys`, um die zulässigen Tag-Schlüssel zu spezifizieren, z. B. `Owner` oder `CreationDate`. Weitere Informationen finden Sie unter Erstellen einer Bedingung, die mehrere Schlüsselwerte testet im IAM-Benutzerhandbuch.

Die Richtlinie stellt sicher, dass jeder in der Anfrage angegebene Tag-Schlüssel ein autorisierter Tag-Schlüssel ist. Der `ForAnyValue`-Qualifizierer in der Bedingung stellt sicher, dass mindestens einer der spezifizierten Schlüssel in der Anfrage enthalten ist.

```
{
```

```
   "Version": "2012-10-17",
  "Statement": [
    {"Principal":{"AWS":[
            "arn:aws:iam::111122223333:role/JohnDoe"
         ]
       },
 "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"ForAnyValue:StringEquals": {"s3:RequestObjectTagKeys": [
            "Owner",
            "CreationDate"
          ]
        }
      }
    }
  ]
}
```

Example 3: Einen spezifischen Tag-Schlüssel und -Wert verlangen, wenn Benutzern erlaubt wird, Objekt-Tags hinzuzufügen

Die folgende Beispielrichtlinie erteilt einem Benutzer die Berechtigungen, die `s3:PutObjectTagging`-Aktion auszuführen, die dem Benutzer gestattet, einem vorhandenen Objekt Markierungen hinzuzufügen. Die Bedingung verlangt, dass der Benutzer einen spezifischen Tag-Schlüssel (z. B. *`Project`*) mit dem Wert *`X`* angibt.

```
{
   "Version": "2012-10-17",
  "Statement": [
    {"Principal":{"AWS":[
       "arn:aws:iam::111122223333:user/JohnDoe"
        ]
       },
      "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
```

```
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"StringEquals": {"s3:RequestObjectTag/Project": "X"
        }
      }
    }
  ]
}
```

Verwalten von Objekt-Markierungen

In diesem Abschnitt wird erläutert, wie Sie Objekt-Markierungen mithilfe der AWS-SDKs für Java und .NET oder der Amazon-S3-Konsole verwalten können.

Das Markieren von Objekten ermöglicht Ihnen, Speicher zu kategorisieren. Jedes Tag ist ein Schlüssel-Wert-Paar, für das folgende Regeln gelten:

- Sie können einem Objekt bis zu 10 Markierungen zuordnen. Einem Objekt zugeordnete Markierungen müssen eindeutige Tag-Schlüssel haben.
- Ein Tag-Schlüssel kann maximal 128 Unicode-Zeichen lang sein, und die Tag-Werte können bis zu 256 Unicode-Zeichen lang sein. Amazon-S3-Objekt-Tags werden intern in UTF-16 dargestellt. Beachten Sie, dass Zeichen in UTF-16 entweder 1 oder 2 Zeichenpositionen einnehmen.
- Bei Schlüsseln und Werten wird die Groß-/Kleinschreibung berücksichtigt.

Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen. Weitere Informationen zu Tag-Einschränkungen finden Sie unter Einschränkungen benutzerdefinierter Markierungen im AWS Billing and Cost Management Benutzerhandbuch.

Verwenden der S3-Konsole

Hinzufügen von Markern zu einem Objekt

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der die Objekte enthält, denen Sie Markierungen hinzufügen möchten.

   Sie können auch optional zu einem Ordner navigieren.

3. Aktivieren Sie in der Liste Objekte das Kontrollkästchen neben den Namen der Objekte, denen Sie Markierungen hinzufügen möchten.
4. Wählen Sie im Menü Aktionen die Option Markierungen bearbeiten.
5. Überprüfen Sie die aufgelisteten Objekte und wählen Sie Markierungen hinzufügen.
6. Jedes Objekt-Tag ist ein Schlüssel-Wert-Paar. Geben Sie einen Key (Schlüssel) und einen Value (Wert) ein. Um ein weiteres Tag hinzuzufügen, wählen Sie Add Tag (Tag hinzufügen).

   Sie können bis zu 10 Marker für ein Objekt eingeben.
7. Wählen Sie Save Changes (Änderungen speichern).

   Amazon S3 fügt die Markierungen zu den angegebenen Objekten hinzu.

Weitere Informationen finden Sie auch unter Anzeigen von Objekteigenschaften in der Amazon-S3-Konsole und Objekte hochladen in diesem Handbuch.

Verwenden der AWS-SDKs

Java

Das folgende Beispiel veranschaulicht, wie Sie mit dem AWS SDK for Java Markierungen für ein neues Objekt festlegen und Markierungen für ein vorhandenes Objekt abrufen oder ersetzen. Weitere Informationen über das Markieren von Objekten finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.File;
import java.util.ArrayList;
import java.util.List;

public class ManagingObjectTags {
```

```
   public static void main(String[] args) {
       Regions clientRegion = Regions.DEFAULT_REGION;
       String bucketName = "*** Bucket name ***";
       String keyName = "*** Object key ***";
       String filePath = "*** File path ***";

       try {
           AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                   .withCredentials(new ProfileCredentialsProvider())
                   .withRegion(clientRegion)
                   .build();

           // Create an object, add two new tags, and upload the object to Amazon
S3.
           PutObjectRequest putRequest = new PutObjectRequest(bucketName, keyName,
new File(filePath));
           List<Tag> tags = new ArrayList<Tag>();
           tags.add(new Tag("Tag 1", "This is tag 1"));
           tags.add(new Tag("Tag 2", "This is tag 2"));
           putRequest.setTagging(new ObjectTagging(tags));
           PutObjectResult putResult = s3Client.putObject(putRequest);

           // Retrieve the object's tags.
           GetObjectTaggingRequest getTaggingRequest = new
GetObjectTaggingRequest(bucketName, keyName);
           GetObjectTaggingResult getTagsResult =
s3Client.getObjectTagging(getTaggingRequest);

           // Replace the object's tags with two new tags.
           List<Tag> newTags = new ArrayList<Tag>();
           newTags.add(new Tag("Tag 3", "This is tag 3"));
           newTags.add(new Tag("Tag 4", "This is tag 4"));
           s3Client.setObjectTagging(new SetObjectTaggingRequest(bucketName,
keyName, new ObjectTagging(newTags)));
       } catch (AmazonServiceException e) {
           // The call was transmitted successfully, but Amazon S3 couldn't process
           // it, so it returned an error response.
           e.printStackTrace();
       } catch (SdkClientException e) {
           // Amazon S3 couldn't be contacted for a response, or the client
           // couldn't parse the response from Amazon S3.
           e.printStackTrace();
       }
   }
```

```
}
```

.NET

Das folgende Beispiel veranschaulicht, wie Sie mit dem AWS SDK for .NET die Markierungen für ein neues Objekt festlegen und die Markierungen für ein vorhandenes Objekt abrufen oder ersetzen. Weitere Informationen über das Markieren von Objekten finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    public class ObjectTagsTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for the new object ***";
        private const string filePath = @"*** file path ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            PutObjectWithTagsTestAsync().Wait();
        }

        static async Task PutObjectWithTagsTestAsync()
        {
            try
            {
                // 1. Put an object with tags.
```

```
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    FilePath = filePath,
                    TagSet = new List<Tag>{
                        new Tag { Key = "Keyx1", Value = "Value1"},
                        new Tag { Key = "Keyx2", Value = "Value2" }
                    }
                };

                PutObjectResponse response = await
client.PutObjectAsync(putRequest);
                // 2. Retrieve the object's tags.
                GetObjectTaggingRequest getTagsRequest = new GetObjectTaggingRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };

                GetObjectTaggingResponse objectTags = await
client.GetObjectTaggingAsync(getTagsRequest);
                for (int i = 0; i < objectTags.Tagging.Count; i++)
                    Console.WriteLine("Key: {0}, Value: {1}",
objectTags.Tagging[i].Key, objectTags.Tagging[i].Value);


                // 3. Replace the tagset.

                Tagging newTagSet = new Tagging();
                newTagSet.TagSet = new List<Tag>{
                    new Tag { Key = "Key3", Value = "Value3"},
                    new Tag { Key = "Key4", Value = "Value4" }
                };


                PutObjectTaggingRequest putObjTagsRequest = new
PutObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                    Tagging = newTagSet
                };
```

```
                PutObjectTaggingResponse response2 = await
 client.PutObjectTaggingAsync(putObjTagsRequest);

                // 4. Retrieve the object's tags.
                GetObjectTaggingRequest getTagsRequest2 = new
 GetObjectTaggingRequest();
                getTagsRequest2.BucketName = bucketName;
                getTagsRequest2.Key = keyName;
                GetObjectTaggingResponse objectTags2 = await
 client.GetObjectTaggingAsync(getTagsRequest2);
                for (int i = 0; i < objectTags2.Tagging.Count; i++)
                    Console.WriteLine("Key: {0}, Value: {1}",
 objectTags2.Tagging[i].Key, objectTags2.Tagging[i].Value);

            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an
 object"
                        , e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Encountered an error. Message:'{0}' when writing an object"
                    , e.Message);
            }
        }
    }
}
```

Verwenden von Kostenzuordnungs-Markierungen für S3-Buckets

Um die Speicherkosten oder andere Kriterien für einzelne Projekte oder Gruppen von Projekten nachzuverfolgen, kennzeichnen Sie Ihre Amazon S3-Buckets unter Verwendung der Kostenzuordnungs-Markierungen. Ein Kostenzuordnungs-Tag ist ein Schlüssel-Wert-Paar, das Sie definieren und mit einem S3-Bucket verknüpfen. Nachdem Sie Kostenzuordnungs-Markierungen aktiviert haben, verwendet AWS die Markierungen, um Ihre Ressourcenkosten in Ihrem Kostenzuordnungsbericht zu gruppieren. Kostenzuordnungs-Markierungen können nur zur Kennzeichnung von Buckets verwendet werden. Weitere Informationen zu Markierungen, die zur

Kennzeichnung von Objekten verwendet werden, finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

Der Kostenzuordnungsbericht listet die AWS -Nutzung für Ihr Konto nach Produktkategorie und verbundenem Kontobenutzer auf. Der Bericht enthält die gleichen Einzelposten wie der detaillierte Fakturierungsbericht (siehe Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3), und zusätzliche Spalten für Ihre Tag-Schlüssel.

AWS bietet zwei Arten von Kostenzuordnungs-Tags, einen AWS-generierten Tag und benutzerdefinierte Tags. AWS definiert, erstellt und wendet den AWS-generierten `createdBy`-Tag nach einem Amazon S3-CreateBucket-Ereignis für Sie an. Sie definieren, erstellen und verwenden benutzerdefinierte Markierungen für Ihren S3-Bucket.

Sie müssen beide Tag-Typen in der Fakturierungs- und Kostenverwaltungskonsole separat aktivieren, damit sie in Ihren monatlichen Berichten angezeigt werden können. Weitere Informationen über von AWS generierte Markierungen finden Sie unter Von AWS generierte Kostenzuordnungs-Markierungen.

- Informationen zum Erstellen von Markierungen in der Konsole finden Sie unter Anzeigen der Eigenschaften eines S3-Buckets.
- Informationen zum Erstellen von Markierungen mit der Amazon S3-API finden Sie unter PUT Bucket Tagging in der Amazon Simple Storage Service API-Referenz.
- Informationen zum Erstellen von Markierungen mit der AWS CLI finden Sie unter put-bucket-tagging in der AWS CLI-Befehlsreferenz.
- Weitere Informationen zur Aktivierung von Markierungen finden Sie unter Using cost allocation tags (Verwendung von Kostenzuordnungs-Markierungen) im AWS Billing-Benutzerhandbuch.

Benutzerdefinierte Kostenzuordnungs-Markierungen

Ein benutzerdefiniertes Kostenzuordnungs-Tag umfasst folgende Komponenten:

- Der Tag-Schlüssel. Der Tag-Schlüssel ist der Name des Markierungen. Im Tag project/Trinity beispielsweise ist project der Schlüssel. Die Tag-Schlüssel ist eine Zeichenfolge, in der zwischen Groß- und Kleinschreibung unterschieden wird und die 1 bis 128 Unicode-Zeichen enthalten kann.

- Der Tag-Wert. Der Tag-Wert ist eine erforderliche Zeichenfolge. Im Tag project/Trinity beispielsweise ist Trinity der Wert. Die Tag-Wert ist eine Zeichenfolge, in der zwischen Groß- und Kleinschreibung unterschieden wird und die 0 bis 256 Unicode-Zeichen enthalten kann.

Weitere Informationen zu den zulässigen Zeichen für benutzerdefinierte Markierungen und anderen Einschränkungen finden Sie unter User-Defined Tag Restrictions (Einschränkungen benutzerdefinierter Markierungen) im AWS Billing-Benutzerhandbuch. Weitere Informationen über benutzerdefinierte Markierungen finden Sie unter User-Defined Cost Allocation Markierungen (Benutzerdefinierte Kostenzuordnungs-Markierungen) im AWS Billing-Benutzerhandbuch.

S3-Bucket-Markierungen

Jeder S3-Bucket verfügt über einen Tag-Satz. Ein Tag-Satz enthält alle Markierungen, die diesem Bucket zugewiesen sind. Ein Tag-Satz kann bis zu 50 Markierungen enthalten oder leer sein. Schlüssel müssen eindeutig innerhalb eines Tag-Satzes sein, aber die Werte in einem Tag-Satz müssen nicht eindeutig sein. Sie können beispielsweise denselben Wert in Tag-Sätzen mit den Namen project/Trinity und cost-center/Trinity haben.

Wenn Sie in einem Bucket ein Tag mit demselben Schlüssel wie für ein vorhandenes Tag hinzufügen, wird der alte Wert mit dem neuen überschrieben.

AWS ordnet Ihren Markierungen keine semantische Bedeutung zu. Wir interpretieren Markierungen streng als Zeichenfolgen.

Um Markierungen hinzuzufügen, zu bearbeiten oder zu löschen, verwenden Sie die Amazon S3-Konsole, die AWS Command Line Interface (AWS CLI) oder die Amazon S3-API.

Weitere Infos

- Verwenden von Kostenzuordnungs-Markierungen im AWS Billing-Benutzerhandbuch.
- Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3
- AWS Billing -Berichte für Amazon S3

Fakturierungs- und Nutzungsberichte für Amazon S3

Wenn Sie Amazon S3 verwenden, müssen Sie keine Vorabgebühren zahlen oder sich darauf festlegen, wie viele Inhalte Sie speichern werden. Wie bei anderen zahlen Sie unverändert und nur für das AWS-Services, was Sie tatsächlich nutzen.

AWS stellt die folgenden Berichte für Amazon S3 bereit:

- Fakturierungsberichte – Mehrere Berichte, die einen allgemeinen Überblick über alle Aktivitäten für die bieten AWS-Services , die Sie verwenden, einschließlich Amazon S3. berechnet dem

Eigentümer des S3-Buckets AWS immer Amazon S3-Gebühren, es sei denn, der Bucket wurde als Bucket mit Zahlung durch den Anforderer erstellt. Weitere Informationen über Zahlung durch den Anforderer finden Sie unter Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung. Weitere Informationen über Fakturierungsberichte finden Sie unter AWS Billing -Berichte für Amazon S3.

- Nutzungsbericht – Eine Zusammenfassung von Aktivitäten für einen bestimmten Service, zusammengefasst nach Stunde, Tag und Monat. Sie können wählen, welche Nutzungsart und welche Operation aufgenommen werden soll. Sie können auch festlegen, wie die Daten aggregiert werden. Weitere Informationen finden Sie unter AWS -Nutzungsbericht für Amazon S3.

Die folgenden Themen enthalten Informationen zur Fakturierung und zu Nutzungsberichten für Amazon S3.

Themen

AWS Billing -Berichte für Amazon S3

Ihre monatliche Rechnung von AWS trennt Ihre Nutzungsinformationen und -kosten nach AWS-Service und Funktion. Es sind mehrere AWS Billing Berichte verfügbar: der monatliche Bericht, der Kostenzuordnungsbericht und detaillierte Fakturierungsberichte. Informationen zum Anzeigen Ihrer Fakturierungsberichte finden Sie unter Viewing Your Bill (Anzeigen Ihrer Rechnung) im AWS Billing - Benutzerhandbuch.

Um Ihre AWS Nutzung zu verfolgen und geschätzte Gebühren für Ihr Konto anzugeben, können Sie einrichten AWS Cost and Usage Reports. Weitere Informationen finden Sie unter Was sind AWSCost and Usage Reports? im AWS Leitfaden für Datenexporte.

Sie können auch einen Nutzungsbericht herunterladen, der Ihnen weitere Informationen über Ihre Amazon-S3-Nutzung mitteilt, als die Fakturierungs-Berichte. Weitere Informationen finden Sie unter AWS -Nutzungsbericht für Amazon S3.

Die folgende Tabelle listet die Gebühren für die Amazon-S3-Nutzung auf.

Amazon-S3-Nutzungsgebühren

Gebühr	Kommentare
Speicher	Sie zahlen für das Speichern von Objekten in S3-Buckets. Der Ihnen in Rechnung gestellte Preis hängt von der Größe Ihrer Objekte, der Dauer der Speicherung der Objekte im Monat und der Speicherklasse ab. Amazon S3 bietet die folgenden Speicherklassen: S3 Standard, S3 Express One Zone, S3 Intelligent-Tierin g, S3 Standard-IA (IA für seltenen Zugriff), S3 One Zone-IA, S3 Glacier Instant Retrieval , S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive oder Reduced Redundancy Storage (RRS). Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen. Beachten Sie, dass Ihnen bei aktiviertem S3-Versioning jede beibehaltene Version eines Objekts in Rechnung gestellt wird. Weitere Informationen über das Versioning finden Sie unter Wie funktioniert S3-Versioning.
Überwachen und Automatisieren	Für jedes in der Speicherklasse S3 Intelligent-Tiering gespeicherte Objekt zahlen Sie eine monatliche Überwachungs- und Automatis ierungsgebühr zur Überwachung der Zugriffsm uster und Verschiebung der Objekte zwischen den Zugriffsstufen des S3 Intelligent-Tiering.
Anforderungen	Sie zahlen für Anforderungen, z. B. `GET` für Anforderungen, die an Ihre S3-Buckets und -Objekte gestellt werden. Dazu zählen auch Lebenszyklus-Anfragen. Die Preise für Anfragen hängen davon ab, welche Art von Anfrage Sie stellen. Weitere Informationen zu

Gebühr	Kommentare
	den Anfragepreisen erhalten Sie unter Amazon-S3-Preise.
Abrufe	Sie zahlen für das Abrufen von Objekten, die sich im Speicher S3 Standard-IA, S3 One Zone-IA-, S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive befinden.
Frühe Löschungen	Wenn Sie ein Objekt im Speicher S3 Standard-IA, S3 One Zone-IA, S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive löschen, bevor die minimale Speicherzusage abgelaufen ist, zahlen Sie eine Gebühr für das vorzeitige Löschen für dieses Objekt.
Speicherverwaltung	Sie zahlen für die Speicherverwaltungsfunktionen (Amazon S3 Inventory, Analysen und Objektmarkierung), die in den Buckets Ihres Kontos aktiviert sind.

Gebühr	Kommentare
Bandbreite	Sie zahlen für die gesamte Bandbreite in und aus Amazon S3 mit Ausnahme des Folgenden: • Datenübertragung aus dem Internet • Daten, die an eine Amazon Elastic Compute Cloud (Amazon EC2)-Instance übertragen werden, wenn sich die Instance in derselben AWS-Region wie der S3-Bucket befindet • Daten, die nach Amazon CloudFront (CloudFront) übertragen werden Sie zahlen auch eine Gebühr für alle Daten, die mit Amazon S3 Transfer Acceleration übertragen werden.

Ausführliche Informationen zu Amazon S3-Nutzungsgebühren für Speicher, Datenübertragung und Services finden Sie unter Amazon S3-Preise und Häufig FAQs zu Amazon S3.

Informationen zum Verständnis der Codes und Abkürzungen, die in den Fakturierungs- und Nutzungsberichten für Amazon S3 verwendet werden, finden Sie unter Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3.

Weitere Informationen

- AWS -Nutzungsbericht für Amazon S3
- Verwenden von Kostenzuordnungs-Markierungen für S3-Buckets
- AWS Billing und Kostenmanagement
- Amazon S3 – Preise

AWS -Nutzungsbericht für Amazon S3

Wenn Sie einen Nutzungsbericht herunterladen, können Sie die Nutzungsdaten nach Stunde, Tag und Monat zusammenfassen. Der Amazon S3-Nutzungsbericht listet Vorgänge nach Nutzungstyp und auf AWS-Region. Für detailliertere Nutzungsberichte über Ihre Amazon-S3-Speichernutzung laden Sie dynamisch generierte AWS -Nutzungsberichte herunter. Sie können wählen, welche Nutzungsart, welche Operation und welcher Zeitraum aufgenommen werden soll. Sie können auch festlegen, wie die Daten aggregiert werden. Weitere Informationen zu Nutzungsberichten finden Sie unter AWS Nutzungsbericht im AWS Benutzerhandbuch für Datenexporte.

Der Amazon-S3-Nutzungsbericht enthält die folgenden Informationen:

- Service – Amazon S3
- Operation – Die Operation, die für Ihren Bucket oder Ihr Objekt durchgeführt wird. Eine detaillierte Erklärung der Amazon-S3-Vorgänge finden Sie unter Nachverfolgen von Vorgängen in Ihren Nutzungsberichten.
- UsageType – Einer der folgenden Werte:
  - Ein Code zur Identifizierung des Speichertyps
  - Ein Code zur Identifizierung der Anfrage
  - Ein Code zur Identifizierung des Abruftyps
  - Ein Code zur Identifizierung des Typs der Datenübertragung
  - Ein Code, der vorzeitiges Löschen aus dem S3 Intelligent-Tiering, S3 Standard-IA-, S3-One-Zone-Infrequent-Access (S3 One Zone-IA)-, S3 Glacier Flexible Retrieval- oder S3 Glacier Deep Archive Speicher identifiziert
  - `StorageObjectCount` – Die Anzahl der Objekte in einem bestimmten Bucket

  Eine detaillierte Erklärung der Amazon-S3-Nutzungstypen finden Sie unter Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3.
- Resource (Ressource) – Der Name des Buckets, der der aufgelisteten Nutzung zugeordnet ist.
- StartTime – Startzeit des Tages, für den die Nutzung gilt, in koordinierter Weltzeit (UTC).
- EndTime – Endzeit des Tages, für den die Nutzung gilt, in koordinierter Weltzeit (UTC).
- UsageValue – Einer der folgenden Volume-Werte. Die typische Maßeinheit für Daten ist Gigabyte (GB). Abhängig vom Dienst und dem Bericht können stattdessen Terabytes (TB) angezeigt werden.
  - Die Anzahl der Anforderungen in dem angegebenen Zeitraum

- Die übertragene Datenmenge
- Die Menge der in einer bestimmten Stunde gespeicherten Daten
- Die Menge der Daten im Zusammenhang mit Wiederherstellungen aus dem Speicher S3 Standard-IA, S3 One Zone-IA, S3 Glacier Flexible Retrieval-, oder S3 Glacier Deep Archive

> **Tip**
>
> Detaillierte Informationen über alle Anfragen, die Amazon S3 für Ihre Objekte empfängt, aktivieren Sie die Server-Zugriffsprotokollierung für Ihre Buckets Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

Sie können einen Nutzungsbericht als XML- oder CSV-Datei herunterladen. Das folgende Beispiel zeigt einen CSV-Nutzungsbericht, der in einer Tabellenkalkulation geöffnet ist.

Service	Operation	UsageType	Resource	StartTime	EndTime	UsageValue
AmazonS3	HeadBucket	USW2-C3DataTransfer-Out-Bytes	admin-created3	6/1/2017 0:00	7/1/2017 0:00	15309
AmazonS3	PutObject	USW2-C3DataTransfer-In-Bytes	admin-created3	6/1/2017 0:00	7/1/2017 0:00	19062
AmazonS3	HeadBucket	USW2-Requests-Tier2	admin-created3	6/1/2017 0:00	7/1/2017 0:00	68
AmazonS3	PutObjectForRepl	USW1-Requests-SIA-Tier1	ca-example-bucket	6/1/2017 0:00	7/1/2017 0:00	178294
AmazonS3	PutObjectForRepl	USW1-USW2-AWS-In-Bytes	ca-example-bucket	6/1/2017 0:00	7/1/2017 0:00	387929083
AmazonS3	GetObjectForRepl	USW2-Requests-NoCharge	admin-created3	6/1/2017 0:00	7/1/2017 0:00	108
AmazonS3	GetObjectForRepl	USW2-USW1-AWS-Out-Bytes	my-test-bucket-bash	6/1/2017 0:00	7/1/2017 0:00	387910021

Weitere Informationen finden Sie unter Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3.

Herunterladen des AWS Nutzungsberichts

Sie können einen Nutzungsbericht als XML- oder CSV-Datei herunterladen.

Den Nutzungsbericht herunterladen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Titelleiste Ihren Benutzernamen oder Ihre Konto-ID und dann Fakturierung und Kostenmanagement aus.
3. Wählen Sie im Navigationsbereich Kosten- und Nutzungsberichte aus.
4. Wählen Sie unter AWS Nutzungsbericht die Option Nutzungsbericht erstellen aus.
5. Wählen Sie auf der Seite Nutzungsbericht herunterladen die folgenden Einstellungen aus:

- Services – Wählen Sie Amazon Simple Storage Service aus.
- Usage Types (Nutzungstypen) – Eine detaillierte Erläuterung der Amazon-S3-Nutzungstypen finden Sie unter Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3.
- Operation – Eine detaillierte Erläuterung der Amazon-S3-Vorgänge finden Sie unter Nachverfolgen von Vorgängen in Ihren Nutzungsberichten.
- Time Period (Zeitraum) – Der Zeitraum, der in dem Bericht erfasst werden soll.
- Report Granularity (Berichtsgranularität) – Gibt an, ob in dem Bericht Zwischensummen nach Stunde, Tag oder Monat angezeigt werden sollen.

6. Wählen Sie Herunterladen, wählen Sie das Download-Format (XML-Bericht oder CSV-Bericht) und folgen Sie dann den Anweisungen, um den Bericht zu öffnen oder zu speichern.

Weitere Informationen

- Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3
- AWS Billing -Berichte für Amazon S3

Grundlegendes zu Ihren AWS Fakturierungs- und Nutzungsberichten für Amazon S3

Amazon-S3-Fakturierungs- und Nutzungsberichte verwenden Codes und Abkürzungen. Ersetzen Sie für Verwendungstypen in der folgenden Tabelle *`regionregion1`*, und *`region2`* durch Abkürzungen aus dieser Liste:

- APE1: Asien-Pazifik (Hongkong)
- APN1: Asien-Pazifik (Tokio)
- APN2 Asien-Pazifik (Seoul)
- APN3: Asien-Pazifik (Osaka)
- APS1: Asien-Pazifik (Singapur)
- APS2: Asien-Pazifik (Sydney)
- APS3: Asien-Pazifik (Mumbai)
- APS4: Asien-Pazifik (Jakarta)
- APS5: Asien-Pazifik (Hyderabad)

- APS6: Asien-Pazifik (Melbourne)
- CAN1: Kanada (Zentral)
- CNN1: China (Peking)
- CNW1: China (Ningxia)
- AFS1: Afrika (Kapstadt)
- EUC2: Europa (Zürich)
- EUN1: Europa (Stockholm)
- EUS2: Europa (Spanien)
- EUC1: Europa (Frankfurt)
- EU: Europa (Irland)
- EUS1: Europa (Mailand)
- EUW2: Europa (London)
- EUW3: Europa (Paris)
- ILC1: Israel (Tel Aviv)
- MEC1: Naher Osten (VAE)
- MES1: Naher Osten (Bahrain)
- SAE1: Südamerika (São Paulo)
- UGW1: AWS GovCloud (USA-West)
- UGE1: AWS GovCloud (USA-Ost)
- USE1 (oder kein Präfix): USA Ost (Nord-Virginia)
- USE2: USA Ost (Ohio)
- USW1: USA West (Nordkalifornien)
- USW2: USA West (Oregon)

Ersetzen Sie für die Verwendungstypen von S3 Multi-Region Access Points in der folgenden Tabelle *`regiongroup1`* und durch *`regiongroup2`* Abkürzungen aus dieser Liste:

- AP: Asien-Pazifik
- AU: Australien
- EU: Europa
- IN: Indien

- NA: Nordamerika
- SA: Südamerika

Regionsgruppen sind geografische Gruppierungen mehrerer AWS-Regionen. Weitere Informationen finden Sie unter Regionen und Availability Zones. Weitere Informationen zu den Preisen nach AWS-Region finden Sie unter Amazon S3 – Preise.

Die erste Spalte in der folgenden Tabelle listet die Nutzungstypen auf, die in Ihren Fakturierungs- und Nutzungsberichten vorkommen. Die typische Maßeinheit für Daten ist Gigabyte (GB). Abhängig vom Dienst und dem Bericht können stattdessen Terabytes (TB) angezeigt werden.

Verwendungstypen

Verwendungstyp	Einheiten	Granularity	Beschreibung
region1-*region2*-AWS-In-A Bytes	GB	Stundensatz	Die Menge der von in übertragenen *region1* beschleunigten Daten *region2*
region1-*region2*-AWS-In-A Bytes-T1	GB	Stundensatz	Die Menge der T1-beschleunigten Daten*region2*, die *region1* von nach übertragen werden, wobei sich T1 auf CloudFront Anfragen an Points of Presence (POPs) in den USA, Europa und Japan bezieht
region1-*region2*-AWS-In-A Bytes-T2	GB	Stundensatz	Die Menge der T2-beschleunigten Daten*region2*, die *region1* von in übertragen werden, wobei sich T2 auf CloudFront Anforderungen an POPs an an allen anderen AWS Edge-Standorten bezieht

Verwendungstyp	Einheiten	Granularity	Beschreibung
region1-*region2*-AWS-In-Bytes	GB	Stundensatz	Die Menge der Daten, die von in übertragen *region1* werden *region2*
region1-*region2*-AWS-Out-ABytes	GB	Stundensatz	Die Menge der von *region1* nach übertragenen beschleunigten Daten *region2*
region1-*region2*-AWS-Out-ABytes-T1	GB	Stundensatz	Die Menge der T1-beschleunigten Daten*region2*, die *region1* von übertragen werden, wobei sich T1 auf CloudFront Anfragen an POPs in den USA, Europa und Japan bezieht
region1-*region2*-AWS-Out-ABytes-T2	GB	Stundensatz	Die Menge der T2-beschleunigten Daten*region2*, die von *region1* nach übertragen werden, wobei sich T2 auf CloudFront Anfragen an POPs in allen anderen AWS Edge-Standorten bezieht
region1-*region2*-AWS-Out-Bytes	GB	Stundensatz	Die Menge der von *region1* nach übertragenen Daten *region2*
region-BatchOperations-Jobs	Anzahl	Stundensatz	Die Anzahl der ausgeführten S3-Batchvorgangsaufträge

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-BatchOperations-Objects	Anzahl	Stundensatz	Die Anzahl der von S3-Batchvorgänge ausgeführten Objektvorgänge
region-Bulk-Retrieval-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Massenanforderungen von S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive abgerufen wurden
region-BytesDeleted-GDA	GB	Monatlich	Die Menge der Daten, die durch eine DeleteObject Operation aus dem Speicher S3 Glacier Deep Archive gelöscht wurden
region-BytesDeleted-GIR	GB	Monatlich	Die Menge der Daten, die durch eine DeleteObject Operation aus dem Speicher S3 Glacier Instant Retrieval gelöscht wurden.
region-BytesDeleted-GLACIER	GB	Monatlich	Die Menge der Daten, die durch eine DeleteObject Operation aus dem Speicher S3 Glacier Flexible Retrieval gelöscht wurden
region-BytesDeleted-INT	GB	Monatlich	Die Menge der Daten, die durch eine DeleteObject Operation aus dem S3-Intelligent-Tiering-Speicher gelöscht wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-BytesDeleted-RRS	GB	Monatlich	Die Menge der Daten, die durch eine DeleteObject Operation aus dem Speicher von Reduced Redundancy Storage (RRS) gelöscht wurden
region-BytesDeleted-SIA	GB	Monatlich	Die Menge der Daten, die durch eine DeleteObject Operation aus dem S3-Standard-IA-Speicher gelöscht wurden
region-BytesDeleted-STANDARD	GB	Monatlich	Die Menge der Daten, die von einer DeleteObject Operation aus dem S3-Standard-Speicher gelöscht wurden
region-BytesDeleted-ZIA	GB	Monatlich	Die Menge der Daten, die durch eine DeleteObject Operation aus dem S3-One-Zone-IA-Speicher gelöscht wurden
region-C3DataTransfer-In-Bytes	GB	Stundensatz	Die Menge der Daten, die von Amazon EC2 innerhalb derselben in Amazon S3 übertragen werden AWS-Region
region-C3DataTransfer-Out-Bytes	GB	Stundensatz	Die Menge der Daten, die von Amazon S3 an Amazon EC2 innerhalb derselben AWS-Regionübertragen wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-CloudFront-In-Bytes	GB	Stundensatz	Die Menge der Daten, die AWS-Region aus einer -Verteilung in einen CloudFront übertragen werden
region-CloudFront-Out-Bytes	GB	Stundensatz	Die Menge der von einem AWS-Region in eine CloudFront Verteilung übertragenen Daten
region-DataTransfer-In-Bytes	GB	Stundensatz	Die Menge der aus dem Internet zu Amazon S3 übertragenen Daten
region-DataTransfer-Out-Bytes	GB	Stundensatz	Die Menge der von Amazon S3 ins Internet übertragenen Daten[1]
region-DataTransfer-Regional-Bytes	GB	Stundensatz	Die Menge der Daten, die von Amazon S3 in AWS Ressourcen innerhalb derselben übertragen werden AWS-Region
region-EarlyDelete-ByteHrs	GB-Stunden	Stündlich	Anteilige Speichernutzung für Objekte, die aus dem Speicher S3 Glacier Flexible Retrieval gelöscht wurden, bevor die 90-tägige Mindestnutzung endete [2]

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-EarlyDelete-GDA	GB-Stunden	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem S3-Glacier-Deep-Archive-Speicher gelöscht wurden, bevor die 180-tägige Mindestnutzung endete [2]
region-EarlyDelete-GIR	GB-Stunden	Stündlich	Anteilige Speichernutzung für Objekte, die aus dem Speicher S3 Glacier Instant Retrieval gelöscht wurden, bevor die 90-tägige Mindestnutzung endete.
region-EarlyDelete-GIR-SmObjects	GB-Stunden	Stündlich	Anteilige Speichernutzung für kleine Objekte (kleiner als 128 KB), die aus S3 Glacier Instant Retrieval gelöscht wurden, bevor die 90-tägige Mindestnutzung endete.
region-EarlyDelete-SIA	GB-Stunden	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem S3-Standard-IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endet[3]

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-EarlyDelete-SIA-SmObjects	GB-Stunden	Stundensatz	Anteilige Speichernutzung für kleine Objekte (kleiner als 128 KB), die aus dem S3-Standard-IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endete[3]
region-EarlyDelete-ZIA	GB-Stunden	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem S3-One-Zone-IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endet[3]
region-EarlyDelete-ZIA-SmObjects	GB-Stunden	Stundensatz	Anteilige Speichernutzung für kleine Objekte (kleiner als 128 KB), die aus dem S3-One-Zone-IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endete[3]
region-Expedited-Retrieval-Bytes	GB	Stundensatz	Die Menge der Daten, die mit beschleunigten Anforderungen von S3 Glacier Flexible Retrieval abgerufen wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-Inventory-Objects Listed	Objekte	Stundensatz	Die Anzahl der Objekte, die für eine Objektgruppe (Objekte werden nach Bucket oder Präfix gruppiert) in einer Lagerbestandsliste aufgelistet werden
region-Monitoring-Automation-INT	Objekte	Stundensatz	Die Anzahl an eindeutigen Objekten, die in der S3-Intelligent-Tiering-Speicherklasse überwacht und automatisch abgestuft werden
region-MRAP-Out-Bytes	GB	Stundensatz	Die Menge der Daten, die über einen Endpunkt von S3 Multi-Region Access Points aus Buckets in einer Region übertragen werden (Preise fürMRAP-Datenweiterleitung).
region-MRAP-In-Bytes	GB	Stundensatz	Die Menge der Daten, die über einen Endpunkt von S3 Multi-Region Access Points aus Buckets in einer Region übertragen werden (Preise fürMRAP-Datenweiterleitung).

Verwendungstyp	Einheiten	Granularity	Beschreibung
regiongroup1-regiongroup2- -MRAP-Out-Bytes	GB	Stundensatz	Die Menge der Daten, die über einen Endpunkt von S3 Multi-Region Access Points von einem Bucket in *regiongroup1* an einen Client in übertragen werden, der sich außerhalb des - AWS Netzwerks *regiongroup2* befindet.
regiongroup1-regiongroup2- -MRAP-In-Bytes	GB	Stundensatz	Die Menge der Daten, die von einem Client in *regiongroup2* außerhalb des - AWS Netzwerks über einen Endpunkt von S3 Multi-Region Access Points in *regiongroup1* einen Bucket in übertragen werden.
region-OverwriteBytes-Copy-GDA	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem Speicher S3 Glacier Deep Archive überschrieben wurden
region-OverwriteBytes-Copy-GIR	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem Speicher S3 Glacier Instant Retrieval überschrieben wurden.

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-OverwriteBytes-Copy-GLACIER	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem Speicher S3 Glacier Flexible Retrieval überschrieben wurden
region-OverwriteBytes-Copy-INT	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem S3-Intelligent-Tiering-Speicher überschrieben wurden
region-OverwriteBytes-Copy-RRS	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem Speicher Reduced Redundancy Storage (RRS) überschrieben wurden
region-OverwriteBytes-Copy-SIA	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem S3-Standard-IA-Speicher überschrieben wurden
region-OverwriteBytes-Copy-STANDARD	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem S3-Standard-Speicher überschrieben wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-OverwriteBytes-Copy-ZIA	GB	Monatlich	Die Menge der Daten, die von einer CopyObject Operation aus dem Speicher S3 One Zone-IA überschrieben wurden
region-OverwriteBytes-Put-GDA	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem Speicher S3 Glacier Deep Archive überschrieben wurden
region-OverwriteBytes-Put-GIR	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem Speicher S3 Glacier Instant Retrieval überschrieben wurden.
region-OverwriteBytes-Put-GLACIER	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem Speicher S3 Glacier Flexible Retrieval überschrieben wurden
region-OverwriteBytes-Put-INT	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem S3-Intelligent-Tiering-Speicher überschrieben wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-OverwriteBytes-Put-RRS	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem Speicher Reduced Redundancy Storage (RRS) überschrieben wurden
region-OverwriteBytes-Put-SIA	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem S3-Standard-IA-Speicher überschrieben wurden
region-OverwriteBytes-Put-STANDARD	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem S3-Standard-Speicher überschrieben wurden
region-OverwriteBytes-Put-ZIA	GB	Monatlich	Die Menge der Daten, die von einer PutObject Operation aus dem S3-One-Zone-IA-Speicher überschrieben wurden
region1-*region2*-S3RTC-In-Bytes	GB	Monatlich	Die Menge der für die S3-Replikationszeitkontrolle (S3 RTC) von *region2* nach übertragenen Daten *region1*

Verwendungstyp	Einheiten	Granularity	Beschreibung
region1-*region2*-S3RTC-Out-Bytes	GB	Monatlich	Die Menge der für die S3-Replikationszeitkontrolle (S3 RTC) von *region1* nach übertragenen Daten *region2*
region-Requests-GDA-Tier1	Anzahl	Stündlich	Die Anzahl der PUT-, -COPY, -CreateMultipartUpload , POST-UploadPart , - oder -CompleteMultipartUpload Anforderungen für Objekte von S3 Glacier Deep Archive [6]
region-Requests-GDA-Tier2	Anzahl	Stündlich	Die Anzahl der - GET und -HEADAnforderungen für Objekte von S3 Glacier Deep Archive
region-Requests-GDA-Tier3	Anzahl	Stundensatz	Die Anzahl der S3 Glacier Deep Archive-Standard-Wiederherstellungsanforderungen
region-Requests-GDA-Tier5	Anzahl	Stundensatz	Die Anzahl der S3 Glacier Deep Archive-Wiederherstellungsmassenanforderungen
region-Requests-GIR-Tier1	Anzahl	Stündlich	Die Anzahl der PUT-COPY, - oder -POSTAnforderungen für Objekte von S3 Glacier Instant Retrieval.

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-Requests-GIR-Tier2	Anzahl	Stündlich	Die Anzahl der GET und aller anderen non-S3-Glacier Instant Retrieval-Tier1-Anforderungen für Objekte von S3 Glacier Instant Retrieval.
region-Requests-GLACIER-Tier1	Anzahl	Stündlich	Die Anzahl der PUT-, -COPY, -POST, CreateMultipartUpload -UploadPart , - oder -CompleteMultipartUpload Anforderungen für Objekte von S3 Glacier Flexible Retrieval [6]
region-Requests-GLACIER-Tier2	Anzahl	Stündlich	Die Anzahl der GET und aller anderen Anforderungen, die nicht auf Objekten von S3 Glacier Flexible Retrieval aufgeführt sind
region-Requests-INT-Tier1	Anzahl	Stündlich	Die Anzahl der PUT-COPY, - oder -POSTAnforderungen für S3-Intelligent-Tiering-Objekte
region-Requests-INT-Tier2	Anzahl	Stündlich	Die Anzahl der GET und aller anderen non-Tier1 -Anforderungen für S3 Intelligent-Tiering-Objekte

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-Requests-SIA-Tier1	Anzahl	Stündlich	Die Anzahl der PUT-COPY, - oder -POSTAnforderungen für S3-Standard-IA-Objekte
region-Requests-SIA-Tier2	Anzahl	Stündlich	Die Anzahl der GET und aller anderen non-S3-Glacier Instant Retrieval-Tier1-Anforderungen für S3-Standard-IA-Objekte
region-Requests-Tier1	Anzahl	Stündlich	Die Anzahl der PUT-COPY, - oder -POSTAnforderungen für S3 Standard-, RRS- und -Tags sowie LIST Anforderungen für alle Buckets und Objekte
region-Requests-Tier2	Anzahl	Stündlich	Die Anzahl der GET und aller anderen non-Tier1-Anforderungen
region-Requests-Tier3	Anzahl	Stündlich	Die Anzahl der Lebenszyklusanforderungen an Wiederherstellungsanforderungen von S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive und Standard S3 Glacier Flexieble Retrieval

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-Requests-Tier4	Anzahl	Stündlich	Die Anzahl der Lebenszyklus-Überg änge in Speicher von S3 Glacier Instant Flexible Retrieval, S3 Intelligent-Tiering, S3 Standard-IA, oder S3 One Zone-IA
region-Requests-Tier5	Anzahl	Stündlich	Die Anzahl der Wiederher stellungsmassenanf orderungen von S3 Glacier Flexible Retrieval
region-Requests-Tier6	Anzahl	Stündlich	Die Anzahl der beschleun igten Wiederherstellungs anforderungen von S3 Glacier Flexible Retrieval
region-Requests-Tier8	Anzahl	Stündlich	Die Anzahl der S3-Access-Grants-A nforderungen
region-Requests-XZ-Tier1	Anzahl	Stündlich	Die Anzahl der - PUT oder -COPYAnforderungen für Objekte von S3 Express One Zone
region-Requests-XZ-Tier2	Anzahl	Stündlich	Die Anzahl der GET und aller anderen non-S3-Express-One-Zone-Tie r1-Anforderungen für Objekte der S3 Express One Zone

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-Requests-ZIA-Tier1	Anzahl	Stündlich	Die Anzahl der PUT-COPY, - oder -POSTAnforderungen für S3-One-Zone-IA-Objekte
region-Requests-ZIA-Tier2	Anzahl	Stündlich	Die Anzahl der GET und aller anderen non-S3-One-Zone-IA-Tier1-Anforderungen für S3-One-Zone-IA-Objekte
region-Retrieval-GIR	GB	Stundensatz	Die Menge der Daten, die aus dem Speicher S3 Glacier Instant Retrieval abgerufen wurden.
region-Retrieval-SIA	GB	Stundensatz	Die Menge der Daten, die aus dem S3-Standard-IA-Speicher abgerufen wurden
region-Retrieval-XZ	GB	Stundensatz	Der Teil der Daten, der 512 KB in einer bestimmten Abrufanforderung (PUT oder COPY) mit S3 Express One Zone-Speicher überschreitet
region-Retrieval-ZIA	GB	Stundensatz	Die Menge der Daten, die aus dem S3-One-Zone-IA Speicher abgerufen wurde

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-S3DSSE-In-Bytes	GB	Monatlich	Die Menge der Daten, die von Amazon S3 doppelt verschlüsselt wurden
region-S3DSSE-Out-Bytes	GB	Monatlich	Die Menge der doppelt verschlüsselten Daten, die von Amazon S3 entschlüsselt wurden
region-S3G-DataTransfer-In-Bytes	GB	Stundensatz	Die Datenmenge, die in Amazon S3 übertragen wird, um Objekte aus dem S3 Glacier Flexible Retrieval- oder S3 Glacier Deep Archive-Speicher wiederherzustellen
region-S3G-DataTransfer-Out-Bytes	GB	Stundensatz	Die Menge der Daten, die von Amazon S3 zum Übergang von Objekten in den Speicher S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive übertragen werden
region-Select-Returned-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-Standard-Speicher zurückgegeben wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-Select-Returned-GIR-Bytes	GB	Stundensatz	Die Datenmenge, die mit Select-Anfragen vom Speicher S3 Glacier Instant Retrieval zurückgegeben wird.
region-Select-Returned-INT-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-Intelligent-Tiering-Speicher zurückgegeben wurden
region-Select-Returned-SIA-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-Standard-IA-Speicher zurückgegeben wurden
region-Select-Returned-ZIA-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-One-Zone-IA-Speicher zurückgegeben wurden
region-Select-Scanned-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-Standard-Speicher gescannt wurden
region-Select-Scanned-GIR-Bytes	GB	Stundensatz	Die Datenmenge, die mit Select-Anfragen aus dem Speicher S3 Glacier Instant Retrieval gescannt wurde.

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-Select-Scanned-INT-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-Intelligent-Tiering-Speicher gescannt wurden
region-Select-Scanned-SIA-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-Standard-IA-Speicher gescannt wurden
region-Select-Scanned-ZIA-Bytes	GB	Stundensatz	Die Menge der Daten, die mit Select-Anforderungen aus dem S3-One-Zone-IA-Speicher gescannt wurden
region-Standard-Retrieval-Bytes	GB	Stundensatz	Die Anzahl der Bytes der Daten, die mit Standardanforderungen von S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive abgerufen wurden
region-StorageAnalytics-ObjCount	Objekte	Stündlich	Die Anzahl der eindeutigen Objekte, die in jeder Storage-Class-Analysis-Konfiguration überwacht werden.

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-StorageLens-ObjCount	Objekte	Täglich	Die Anzahl der eindeutigen Objekte in jedem S3 Storage Lens Dashboard, die durch erweiterte Metriken und Empfehlungen von S3 Storage Lens verfolgt werden.
region-StorageLensFreeTier-ObjCount	Objekte	Täglich	Die Anzahl der eindeutigen Objekte in jedem S3 Storage Lens Dashboard, die durch Nutzungsmetriken von S3 Storage Lens verfolgt werden.
StorageObjectCount	Anzahl	Täglich	Die Anzahl der Objekte in einem bestimmten Bucket
region-TagStorage-TagHrs	Tag-Hours	Täglich	Die Gesamtzahl der Markierungen für alle Objekte im Bucket, gemeldet nach Stunde
region-TimedStorage-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, wie lange die Daten im S3-Standard-Speicher gespeichert wurden
region-TimedStorage-GDA-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die die Daten im S3-Glacier-Deep-Archive-Speicher gespeichert wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-TimedStorage-GDA-Staging	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die die Daten im S3-Glacier-Deep-Archive-Staging-Speicher gespeichert wurden
region-TimedStorage-GIR-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die Daten im Speicher S3 Glacier Instant Retrieval gespeichert wurden.
region-TimedStorage-GIR-SmObjects	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, die kleine Objekte (kleiner als 128 KB) im Speicher S3 Glacier Instant Retrieval gespeichert wurden.
region-TimedStorage-GlacierByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die Daten im Speicher S3 Glacier Flexible Retrieval gespeichert wurden
region-TimedStorage-GlacierStaging	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die Daten im Speicher S3 Glacier Flexible Retrieval gespeichert wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-TimedStorage-INT-FA-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die Daten in der Stufe für häufigen Zugriff des S3-Intelligent-Tiering-Speichers gespeichert wurden[5]
region-TimedStorage-INT-IA-ByteHrs	GB-Stunden	Täglich	Die Anzahl an GB-Stunden, die angibt, wie lange die Daten in der Stufe für seltenen Zugriff im S3-Intelligent-Tiering-Speicher gespeichert wurden
region-TimedStorage-INT-AA-ByteHrs	GB-Stunden	Täglich	Die Anzahl an GB-Stunden, die angibt, wie lange die Daten in der Stufe für Archivzugriff im S3-Intelligent-Tiering-Speicher gespeichert wurden
region-TimedStorage-INT-AIA-ByteHrs	GB-Stunden	Täglich	Die Anzahl an GB-Stunden, die angibt, wie lange die Daten in der Stufe für Archive Instant Access im S3-Intelligent-Tiering-Speicher gespeichert wurden
region-TimedStorage-INT-DAA-ByteHrs	GB-Stunden	Täglich	Die Anzahl an GB-Stunden, die angibt, wie lange die Daten in der Stufe für Deep-Archive-Zugriff im S3-Intelligent-Tiering-Speicher gespeichert wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-TimedStorage-RRS-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, wie lange die Daten im RRS(Reduced Redundancy Storage)-Speicher gespeichert wurden
region-TimedStorage-SIA-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, wie lange die Daten im S3-Standard-IA-Speicher gespeichert wurden
region-TimedStorage-SIA-SmObjects	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, die kleine Objekte (kleiner als 128 KB) im S3-Standard-IA-Speicher [4] gespeichert wurden
region-TimedStorage-XZ-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die die Daten im S3-One-Zone-Speicher gespeichert wurden
region-TimedStorage-ZIA-ByteHrs	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, wie lange die Daten im S3-One-Zone-IA-Speicher gespeichert wurden

Verwendungstyp	Einheiten	Granularity	Beschreibung
region-TimedStorage-ZIA-SmObjects	GB-Stunden	Täglich	Die Anzahl der GB-Stunden, für die kleine Objekte (kleiner als 128 KB) im S3-One-Zone-IA-Speicher gespeichert wurden
region-Upload-XZ	GB	Stundensatz	Die Datenmenge, die 512 KB in einer bestimmten Upload-Anforderung (PUT oder COPY) mit S3 Express One Zone überschreitet

Hinweise

1. Wenn Sie eine Übertragung vor dem Abschluss beenden, kann die Menge der übertragenen Daten die Menge der von Ihrer Anwendung erhaltenen Daten überschreiten. Diese Diskrepanz kann auftreten, weil eine Anforderung zur Beendigung der Übertragung nicht sofort ausgeführt werden kann und eine gewisse Datenmenge übertragen werden kann, bis die Beendigungsanforderung ausgeführt wird. Diese Daten während der Übertragung werden als „ausgehende“ Daten abgerechnet.
2. Wenn Objekte, die in den Speicherklassen S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive archiviert werden, gelöscht, überschrieben oder in eine andere Speicherklasse überführt werden, bevor die minimale Speicherzusage abgelaufen ist. Dies entspricht 90 Tagen für S3 Glacier Instant Retrieval und S3 Glacier Flexible Retrieval oder 180 Tagen für S3 Glacier Deep Archive, fällt für die verbleibenden Tage eine anteilige Gebühr pro Gigabyte an.
3. Bei Objekten, die sich im Speicher S3 Standard-IA oder S3 One Zone-IA befinden, fällt für die verbleibenden Tage eine anteilige Gebühr pro Gigabyte an, wenn sie gelöscht, überschrieben oder in eine andere Speicherklasse vor 30 Tagen übertragen werden.
4. Für kleine Objekte (kleiner als 128 KB), die sich im Speicher S3 Standard-IA oder S3 One Zone-IA befinden, fällt bei Löschung, Überschreibung oder Übergang in eine andere Speicherklasse vor Ablauf von 30 Tagen eine anteilige Gebühr pro Gigabyte für die verbleibenden Tage an.

5. In der S3-Intelligent-Tiering-Speicherklasse gibt es keine mindest-abrechnungsfähige Objektgröße. Objekte, die kleiner als 128 KB sind, werden nicht überwacht oder können nicht automatisch gestuft werden. Kleinere Objekte werden immer in der S3-Intelligent-Tiering-Frequent-Access-Stufe gespeichert.
6. Wenn Sie eine `CreateMultipartUpload`-, - `UploadPart`oder -`UploadPartCopy`Anforderung an die Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive initiieren, werden Anforderungen zu den S3-Standardanforderungstarifen abgerechnet, bis Sie den mehrteiligen Upload abgeschlossen haben. Nachdem der Upload abgeschlossen ist, wird die einzelne `CompleteMultipartUpload` Anfrage zum `PUT` Tarif für den S3-Glacier-Zielspeicher abgerechnet. In Bearbeitung befindliche mehrteilige Upload-Teile für einen `PUT` in die Speicherklasse S3 Glacier Flexible Retrieval werden als S3 Glacier Flexible Retrieval Staging Storage zu den S3-Standardspeichertarifen in Rechnung gestellt, bis der Upload abgeschlossen ist. Ebenso werden in Bearbeitung befindliche mehrteilige Upload-Teile für einen `PUT` in die Speicherklasse S3 Glacier Deep Archive als S3 Glacier Deep Archive Staging Storage zu den S3-Standardspeichertarifen in Rechnung gestellt, bis der Upload abgeschlossen ist.
7. S3 Express One Zone erhebt eine pauschale Gebühr pro Anfrage für Anforderungsgrößen bis zu 512 KB. Für `PUT` Anfragen und `GET` Anfragen für den Teil der Anfrage über 512 KB wird eine zusätzliche Gebühr pro GB erhoben.
8. Informationen zu den unterstützten Features der Speicherklasse S3 Express One Zone finden Sie unter Amazon-S3-Features, die von S3 Express One Zone nicht unterstützt werden.
9. Nutzungstypen mit Einheiten, die in GB abgerechnet werden, werden in den Nutzungsberichten in Byte berechnet.

Nachverfolgen von Vorgängen in Ihren Nutzungsberichten

Operationen beschreiben die Aktion, die von dem angegebenen Nutzungstyp an Ihrem AWS Objekt oder Bucket ausgeführt wird. Vorgänge werden durch selbsterklärende Codes beschrieben, z. B. `PutObject` oder `ListBucket`. Um zu prüfen, welche Aktionen für Ihren Bucket einen bestimmten Nutzungstyp generiert haben, verwenden Sie diese Codes. Wenn Sie einen Nutzungsbericht erstellen, können Sie wählen, All Operations aufzunehmen oder eine bestimmte Operation, z. B. `GetObject`, für die ein Bericht erstellt werden soll.

Weitere Informationen

- AWS -Nutzungsbericht für Amazon S3
- AWS Billing -Berichte für Amazon S3

- Amazon S3 – Preise
- Häufig FAQs zu Amazon S3

Filtern und Abrufen von Daten mit Amazon S3 Select

Mit Amazon S3 Select können Sie mit SQL (Structured Query Language)-Anweisungen die Inhalte von Amazon S3-Objekten filtern, sodass nur die tatsächlich von Ihnen benötigte Teilmenge an Daten abgerufen wird. Wenn Sie die Daten mit Amazon S3 Select filtern, wird die von Amazon S3 übertragene Datenmenge reduziert, wodurch sich die Kosten und Latenzzeiten für diesen Datenabruf verringern.

Amazon S3 Select funktioniert für Objekte, die in den Formaten CSV, JSON oder Apache Parquet gespeichert sind. Es funktioniert auch mit Objekten, die mit GZIP oder BZIP2 (nur für CSV- und JSON-Objekte) komprimiert wurden, sowie für serverseitig verschlüsselte Objekte. Als Format für die Ergebnisse geben Sie entweder CSV oder JSON an. Sie können ebenfalls festlegen, wie die Datensätze im Ergebnis getrennt werden.

In der Anforderung werden SQL-Ausdrücke an Amazon S3 übergeben. Amazon S3 Select unterstützt eine Teilmenge von SQL. Weitere Informationen zu den von Amazon S3 Select unterstützten SQL-Elementen finden Sie unter SQL-Referenz für Amazon S3 Select.

Sie können SQL-Abfragen über AWS-SDKs, den `SelectObjectContent`-REST-API-Vorgang, die AWS Command Line Interface (AWS CLI) oder die Amazon S3-Konsole ausführen. Bei der Amazon S3-Konsole ist die zurückgegebene Datenmenge auf 40 MB begrenzt. Sollen mehr Daten abgerufen werden, verwenden Sie die AWS CLI oder die API.

Voraussetzungen und Einschränkungen

Für die Verwendung von Amazon S3 Select müssen folgende Voraussetzungen erfüllt sein:

- Sie haben die Berechtigung `s3:GetObject` für das abzufragende Objekt.
- Falls das abzufragende Objekt serverseitig mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) verschlüsselt ist, müssen Sie `https` verwenden und den Verschlüsselungsschlüssel in der Anforderung angeben.

Für die Verwendung von Amazon S3 Select gelten die folgenden Einschränkungen:

- Die maximale Länge des SQL-Ausdrucks ist 256 KB.

- Die Maximallänge eines Datensatzes in der Eingabe oder im Ergebnis liegt bei 1 MB.
- Amazon S3 Select kann nur mithilfe des JSON-Ausgabeformats verschachtelte Daten übermitteln.
- Sie können Objekte in den Speicherklassen S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive oder Reduced Redundancy Storage (RRS) nicht abfragen. Sie können auch die Objekte auf der S3 Intelligent-Tiering-Zugriffsebene „Archive Access“ oder „Deep Archive“ nicht abfragen. Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.

Bei Verwendung von Amazon S3 Select mit Parquet-Objekten gelten zusätzliche Einschränkungen.

- Amazon S3 Select unterstützt ausschließlich die Spaltenkompression mittels GZIP oder Snappy. Amazon S3 Select unterstützt für Parquet-Objekte keine Kompression ganzer Objekte.
- Amazon S3 Select unterstützt keine Parquet-Ausgabe. Sie müssen das Ausgabeformat als CSV oder JSON angeben.
- Die Maximalgröße für unkomprimierte Zeilengruppen beträgt 512 MB.
- Sie müssen die im Schema des Objekts angegebenen Datentypen verwenden.
- Select-Anfragen für ein wiederholtes Feld geben ausschließlich den letzten Wert zurück.

Erstellen einer Anforderung

Beim Erstellen einer Anforderung geben Sie die Details des abzufragenden Objekts mithilfe eines `InputSerialization`-Objekts an. Die Details zur Rückgabe der Ergebnisse geben Sie mit einem `OutputSerialization`-Objekt an. Außerdem binden Sie den SQL-Ausdruck ein, der von Amazon S3 zum Filtern der Anforderung verwendet wird.

Weitere Informationen zum Erstellen einer Amazon S3 Select-Anforderung finden Sie unter SelectObjectContent in der Referenz zu Amazon Simple Storage Service API. In den folgenden Abschnitten finden Sie eines der SDK-Codebeispiele.

Anforderungen mittels Scanbereichen

Mit Amazon S3 Select können Sie eine Teilmenge eines Objekts scannen, indem Sie einen abzufragenden Byte-Bereich angeben. Mit dieser Fähigkeit können Sie das gesamte Objekt parallel scannen, indem Sie die Arbeit in separate Amazon S3 Select-Anforderungen für eine Reihe von sich nicht überlappenden Scanbereichen aufteilen.

Scanbereiche müssen nicht an Datensatzgrenzen ausgerichtet werden. Eine Scanbereichsanforderung von Amazon S3 Select wird über den angegebenen Byte-Bereich

ausgeführt. Ein Datensatz, der im angegebenen Scanbereich beginnt, aber darüber hinaus reicht, wird von der Abfrage bearbeitet. Beispiel: Im Folgenden wird ein Amazon S3-Objekt gezeigt, das eine Reihe von Datensätzen in einem zeilengetrennten CSV-Format enthält:

```
A,B
C,D
D,E
E,F
G,H
I,J
```

Gehen wir einmal davon aus, dass Sie den Amazon-S3-Select-Parameter `ScanRange` verwenden, bei (Byte) 1 beginnen und bei (Byte) 4 enden. Der Scanbereich würde also bei „`,`“ beginnen und der Scan würde bis zum Ende des Datensatzes fortgesetzt, der bei „`C`“ beginnt. Ihre Scanbereichsanfrage gibt das Ergebnis „`C, D`“ zurück, da dies das Ende des Datensatzes ist.

Scanbereichsanfragen von Amazon S3 Select unterstützen Parquet-, CSV- (ohne Trennung mit Anführungszeichen) und JSON-Objekte (nur im `LINES`-Modus) CSV- und JSON-Objekte dürfen nicht komprimiert sein. Wenn bei zeilenbasierten CSV- und JSON-Objekten ein Scanbereich als Teil der Amazon S3-Select-Anforderung angegeben wird, werden alle Datensätze, die innerhalb des Scanbereichs beginnen, verarbeitet. Bei Parquet-Objekten werden alle Zeilengruppen, die innerhalb des angeforderten Scanbereich beginnen, verarbeitet.

Scanbereichsanfragen von Amazon S3 Select stehen zur Nutzung mit der AWS CLI, der Amazon S3-API und AWS-SDKs zur Verfügung. Sie können den `ScanRange`-Parameter in der Amazon S3 Select-Anfrage als diese Funktion verwenden. Weitere Informationen finden Sie unter SelectObjectContent in der API-Referenz zu Amazon Simple Storage Service.

Fehler

Amazon S3 Select gibt einen Fehlercode und eine zugehörige Fehlermeldung zurück, wenn beim Versuch, eine Abfrage auszuführen, ein Problem auftritt. Eine Liste der Fehlercodes und Beschreibungen finden Sie im Abschnitt List of SELECT Object Content Error Codes auf der Seite Error Responses im Amazon-Simple-Storage-Service-API-Referenz.

Weitere Informationen zu Amazon S3 Select finden Sie in den folgenden Themen:

Themen

- Beispiele für die Verwendung von Amazon S3 Select für Objekte

- SQL-Referenz für Amazon S3 Select

Beispiele für die Verwendung von Amazon S3 Select für Objekte

Sie können S3 Select mit der Amazon S3-Konsole, REST-API und den AWS-SDKs verwenden, um Inhalte aus Objekten auszuwählen.

Weitere Hinweise zu unterstützten SQL-Funktionen für S3 Select finden Sie unter SQL-Funktionen.

Verwenden der S3-Konsole

Inhalte aus einem Objekt in der Amazon S3-Konsole auswählen

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Buckets aus.
3. Wählen Sie den Bucket aus, der das Objekt enthält, aus dem Sie Inhalte auswählen möchten, und wählen Sie dann den Namen des Objekts aus.
4. Wählen Sie Objektaktionen und anschließend Abfragen mit S3 Select aus.
5. Konfigurieren Sie die Eingabeeinstellungen basierend auf dem Format Ihrer Eingabedaten.
6. Konfigurieren Sie die Ausgabeeinstellungen basierend auf dem Format der Ausgabe, die Sie empfangen möchten.
7. Um Datensätze aus dem ausgewählten Objekt zu extrahieren, geben Sie unter SQL-Abfrage die SQL-Befehle SELECT ein. Weitere Informationen zum Schreiben von SQL-Befehlen finden Sie unter SQL-Referenz für Amazon S3 Select.
8. Nachdem Sie SQL-Abfragen eingegeben haben, wählen Sie SQL-Abfrage ausführen aus. Anschließend können Sie unter Abfrageergebnisse die Ergebnisse Ihrer SQL-Abfragen sehen.

Verwenden der REST-API

Mit den AWS-SDKs können Sie Inhalte von Objekten auswählen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Weitere Informationen über das Anforderungs- und Antwort-Format finden sie unter SelectObjectContent.

Verwenden der AWS-SDKs

Sie können Amazon S3 Select nutzen, um Inhalte eines Objekts unter Verwendung der `selectObjectContent`-Methode auszuwählen. Wenn diese Methode erfolgreich ist, gibt sie die Ergebnisse des SQL-Ausdrucks zurück.

Java

Der folgende Java-Code gibt den Wert der ersten Spalte für jeden Datensatz zurück, der in einem Objekt gespeichert ist, das im CSV-Format gespeicherte Daten enthält. Im Beispiel müssen auch `Progress`- und `Stats`-Meldungen zurückgegeben werden. Sie müssen einen gültigen Bucket-Namen sowie ein Objekt angeben, das Daten im CSV-Format enthält.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
package com.amazonaws;

import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CSVInput;
import com.amazonaws.services.s3.model.CSVOutput;
import com.amazonaws.services.s3.model.CompressionType;
import com.amazonaws.services.s3.model.ExpressionType;
import com.amazonaws.services.s3.model.InputSerialization;
import com.amazonaws.services.s3.model.OutputSerialization;
import com.amazonaws.services.s3.model.SelectObjectContentEvent;
import com.amazonaws.services.s3.model.SelectObjectContentEventVisitor;
import com.amazonaws.services.s3.model.SelectObjectContentRequest;
import com.amazonaws.services.s3.model.SelectObjectContentResult;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.concurrent.atomic.AtomicBoolean;

import static com.amazonaws.util.IOUtils.copy;

/**
 * This example shows how to query data from S3Select and consume the response in
 the form of an
 * InputStream of records and write it to a file.
```

```
 */

public class RecordInputStreamExample {

    private static final String BUCKET_NAME = "${my-s3-bucket}";
    private static final String CSV_OBJECT_KEY = "${my-csv-object-key}";
    private static final String S3_SELECT_RESULTS_PATH = "${my-s3-select-results-
path}";
    private static final String QUERY = "select s._1 from S3Object s";

    public static void main(String[] args) throws Exception {
        final AmazonS3 s3Client = AmazonS3ClientBuilder.defaultClient();

        SelectObjectContentRequest request = generateBaseCSVRequest(BUCKET_NAME,
 CSV_OBJECT_KEY, QUERY);
        final AtomicBoolean isResultComplete = new AtomicBoolean(false);

        try (OutputStream fileOutputStream = new FileOutputStream(new File
 (S3_SELECT_RESULTS_PATH));
             SelectObjectContentResult result =
 s3Client.selectObjectContent(request)) {
            InputStream resultInputStream =
 result.getPayload().getRecordsInputStream(
                    new SelectObjectContentEventVisitor() {
                        @Override
                        public void visit(SelectObjectContentEvent.StatsEvent event)
                        {
                            System.out.println(
                                    "Received Stats, Bytes Scanned: " +
 event.getDetails().getBytesScanned()
                                            +  " Bytes Processed: " +
 event.getDetails().getBytesProcessed());
                        }

                        /*
                         * An End Event informs that the request has finished
 successfully.
                         */
                        @Override
                        public void visit(SelectObjectContentEvent.EndEvent event)
                        {
                            isResultComplete.set(true);
                            System.out.println("Received End Event. Result is
 complete.");
```

```
                        }
                    }
            );

            copy(resultInputStream, fileOutputStream);
        }

        /*
         * The End Event indicates all matching records have been transmitted.
         * If the End Event is not received, the results may be incomplete.
         */
        if (!isResultComplete.get()) {
            throw new Exception("S3 Select request was incomplete as End Event was
 not received.");
        }
    }

    private static SelectObjectContentRequest generateBaseCSVRequest(String bucket,
 String key, String query) {
        SelectObjectContentRequest request = new SelectObjectContentRequest();
        request.setBucketName(bucket);
        request.setKey(key);
        request.setExpression(query);
        request.setExpressionType(ExpressionType.SQL);

        InputSerialization inputSerialization = new InputSerialization();
        inputSerialization.setCsv(new CSVInput());
        inputSerialization.setCompressionType(CompressionType.NONE);
        request.setInputSerialization(inputSerialization);

        OutputSerialization outputSerialization = new OutputSerialization();
        outputSerialization.setCsv(new CSVOutput());
        request.setOutputSerialization(outputSerialization);

        return request;
    }
}
```

JavaScript

Ein JavaScript-Beispiel, das AWS SDK for JavaScript mit dem S3 `SelectObjectContent` API-Vorgang zur Auswahl von Datensätzen aus in Amazon S3 gespeicherten JSON- und CSV-

Dateien nutzt, finden Sie im Blogbeitrag Introducing support for Amazon S3 Select in the AWS SDK for JavaScript (Einführung von Support für Amazon S3 Select in…).

Python

Ein Python-Beispiel für die Verwendung von SQL-Abfragen zum Durchsuchen von Daten, die in Amazon S3 als CSV-Datei (Comma-Separated Value) mit S3 Select geladen wurden, finden Sie im Blogbeitrag Querying data without servers or databases using Amazon S3 Select (Daten ohne Server oder Datenbanken mit Amazon S3 Select abfragen).

SQL-Referenz für Amazon S3 Select

In dieser Referenz finden Sie eine Beschreibung der Structured Query Language (SQL)-Elemente, die von Amazon S3 Select unterstützt werden.

Themen

- SELECT command
- Datentypen
- Operatoren
- Reservierte Schlüsselwörter
- SQL-Funktionen

SELECT command

Amazon S3 Select unterstützt nur den SQL-Befehl `SELECT`. Die folgenden ANSI-Standardklauseln werden für `SELECT` unterstützt:

- `SELECT` table
- `FROM`-Klausel
- `WHERE`-Klausel
- `LIMIT`-Klausel

> **Note**
>
> Amazon S3 Select-Abfragen unterstützen derzeit keine Unterabfragen oder Joins.

SELECT table

Die SELECT-Liste nennt die Spalten, Funktionen und Ausdrücke, die die Abfrage zurückgeben soll. Der Liste stellt die Ausgabe der Abfrage dar.

```
SELECT *
SELECT projection1 AS column_alias_1, projection2 AS column_alias_2
```

Bei der ersten SELECT-Form mit dem * (Sternchen) werden alle Zeilen zurückgegeben, die die Bedingung der WHERE-Klausel erfüllen. Bei der zweiten SELECT-Form wird für jede Spalte eine Zeile mit benutzerdefinierten skalaren Ausgabeausdrücken ***projection1*** und ***projection2*** erstellt.

FROM-Klausel

Amazon S3 Select unterstützt die folgenden Formen der FROM-Klausel:

```
FROM table_name
FROM table_name alias
FROM table_name AS alias
```

In jeder Form der FROM-Klausel steht `table_name` für das `S3Object`, das abgefragt wird. Benutzer traditioneller relationaler Datenbanken können sich dies als Datenbankschema mit mehreren Ansichten einer Tabelle vorstellen.

Gemäß Standard-SQL erstellt die FROM-Klausel Zeilen, die in der WHERE-Klausel gefiltert und in der Liste SELECT projiziert werden.

Im Fall von JSON-Objekten, die in Amazon S3 Select gespeichert sind, können Sie auch die folgenden Formate der FROM-Klausel verwenden:

```
FROM S3Object[*].path
FROM S3Object[*].path alias
FROM S3Object[*].path AS alias
```

Unter Verwendung dieses Format der FROM-Klausel können Sie aus Arrays oder Objekten innerhalb eines JSON-Objekts auswählen. Sie können „`path`“ unter Verwendung einer der folgenden Formen angeben:

- Nach Name (in einem Objekt): `.name` oder `['name']`

- Nach Index (in einem Array): `[index]`
- Nach Platzhalterzeichen (in einem Objekt): `.*`
- Nach Platzhalterzeichen (in einem Array): `[*]`

> **Note**
>
> - Das Format der `FROM`-Klausel funktioniert nur mit JSON-Objekten.
> - Platzhalterzeichen übermitteln stets mindestens einen Datensatz. Wenn es keinen übereinstimmenden Datensatz gibt, übermittelt Amazon S3 Select den Wert `MISSING`. Während der Ausgabeserialisierung (nach Abschluss der Abfrage) ersetzt Amazon S3 Select `MISSING`-Werte durch leere Datensätze.
> - Aggregierte Funktionen (`AVG`, `COUNT`, `MAX`, `MIN` und `SUM`) überspringen `MISSING`-Werte.
> - Wenn Sie bei Verwendung eines Platzhalterzeichens keinen Alias angeben, können Sie auf die Zeile verweisen, die das letzte Element im Pfad verwendet. Sie könnten beispielsweise alle Preise aus einer Liste von Büchern unter Verwendung der Abfrage `SELECT price FROM S3Object[*].books[*].price` auswählen. Wenn der Pfad mit einem Platzhalterzeichen anstelle eines Namens endet, können Sie den Wert `_1` verwenden, um auf die Zeile zu verweisen. Beispielsweise könnten Sie anstelle von `SELECT price FROM S3Object[*].books[*].price` die Abfrage `SELECT _1.price FROM S3Object[*].books[*]` verwenden.
> - Amazon S3 Select behandelt ein JSON-Dokument stets als Array aus Werten auf Root-Ebene. Daher muss die `FROM`-Klausel mit `S3Object[*]` beginnen, auch wenn das von Ihnen abgefragte JSON-Objekt nur ein Root-Element hat. Aus Kompatibilitätsgründen ermöglicht Amazon S3 Select Ihnen das Auslassen des Platzhalterzeichens, wenn Sie keinen Pfad einfügen. Daher ist die vollständige Klausel `FROM S3Object` gleichwertig mit `FROM S3Object[*] as S3Object`. Wenn Sie einen Pfad einfügen, müssen Sie auch das Platzhalterzeichen verwenden. Daher sind sowohl `FROM S3Object` als auch `FROM S3Object[*].path` gültige Klauseln, nicht aber `FROM S3Object.path`.

Example

Beispiele:

Beispiel 1

Dieses Beispiel zeigt Ergebnisse unter Verwendung des folgenden Datensatzes und der folgenden Abfrage:

```
{ "Rules": [ {"id": "1"}, {"expr": "y > x"}, {"id": "2", "expr": "z = DEBUG"} ]}
{ "created": "June 27", "modified": "July 6" }
```

```
SELECT id FROM S3Object[*].Rules[*].id
```

```
{"id":"1"}
{}
{"id":"2"}
{}
```

Amazon S3 Select produziert die einzelnen Ergebnisse aus den folgenden Gründen:

- `{"id":"id-1"}` – `S3Object[0].Rules[0].id` produzierte eine Übereinstimmung.
- `{}` – `S3Object[0].Rules[1].id` produzierte keine Übereinstimmung mit einem Datensatz. Daher übermittelte Amazon S3 Select den Wert „`MISSING`“, der anschließend während der Ausgabeserialisierung in einen leeren Datensatz geändert und zurückgegeben wurde.
- `{"id":"id-2"}` – `S3Object[0].Rules[2].id` produzierte eine Übereinstimmung.
- `{}` – `S3Object[1]` produzierte keine Übereinstimmung mit `Rules`. Daher übermittelte Amazon S3 Select den Wert „`MISSING`“, der anschließend während der Ausgabeserialisierung in einen leeren Datensatz geändert und zurückgegeben wurde.

Wenn Sie nicht möchten, dass Amazon S3 Select leere Datensätze zurückgibt, wenn keine Übereinstimmung gefunden wird, können Sie einen Test auf den -Wert ausführe `MISSING`. Die folgende Abfrage gibt dieselben Ergebnisse wie die vorherige Abfrage zurück, jedoch mit Auslassung der leeren Werte:

```
SELECT id FROM S3Object[*].Rules[*].id WHERE id IS NOT MISSING
```

```
{"id":"1"}
{"id":"2"}
```

Beispiel 2

Dieses Beispiel zeigt Ergebnisse unter Verwendung des folgenden Datensatzes und der folgenden Abfragen:

```
{ "created": "936864000", "dir_name": "important_docs", "files": [ { "name": "." },
 { "name": ".." }, { "name": ".aws" }, { "name": "downloads" } ], "owner": "Amazon
 S3" }
{ "created": "936864000", "dir_name": "other_docs", "files": [ { "name": "." },
 { "name": ".." }, { "name": "my stuff" }, { "name": "backup" } ], "owner": "User" }
```

```
SELECT d.dir_name, d.files FROM S3Object[*] d
```

```
{"dir_name":"important_docs","files":[{"name":"."},{"name":".."},{"name":".aws"},
{"name":"downloads"}]}
{"dir_name":"other_docs","files":[{"name":"."},{"name":".."},{"name":"my stuff"},
{"name":"backup"}]}
```

```
SELECT _1.dir_name, _1.owner FROM S3Object[*]
```

```
{"dir_name":"important_docs","owner":"Amazon S3"}
{"dir_name":"other_docs","owner":"User"}
```

WHERE-Klausel

Die `WHERE`-Klausel hat die folgende Syntax:

```
WHERE condition
```

Die `WHERE`-Klausel filtert Zeilen basierend auf den Wert für „*`condition`*“. Eine Bedingung ist ein Ausdruck, der als Ergebnis einen booleschen Wert zurückgibt. Nur Zeilen, deren Bedingung als `TRUE` ausgewertet wird, werden als Ergebnis zurückgegeben.

LIMIT-Klausel

Die `LIMIT`-Klausel hat die folgende Syntax:

```
LIMIT number
```

Die `LIMIT`-Klausel begrenzt die Anzahl der von der Abfrage zurückgegebenen Datensätze basierend auf den Wert für „*`number`*“.

Attributzugriff

Die Klauseln `SELECT` und `WHERE` können mithilfe einer der nachfolgend genannten Methoden auf einen Datensatz verweisen. Das ist abhängig davon, ob die abzufragende Datei im CSV- oder JSON-Format vorliegt.

CSV

- Spaltennummern – Sie können auf die Nte Spalte einer Zeile mit dem Spaltennamen *_N* verweisen, wobei *N* die Spaltenposition angibt. Der Positionszähler beginnt mit 1. Die erste Spalte heißt demnach _1, die zweite Spalte heißt _2.

  Sie können mit *_N* oder *alias*`._`*N* auf eine Spalte verweisen. Beispielsweise sind sowohl _2 und `myAlias._`2 gültige Möglichkeiten, um auf eine Spalte in der `SELECT`-Liste und der `WHERE`-Klausel zu verweisen.
- Spalten-Header – Bei Objekten im CSV-Format, die eine Kopfzeile enthalten, sind die Header für die `SELECT`-Liste und die `WHERE`-Klausel verfügbar. Besonders in `SELECT`- und `WHERE`-Klauselausdrücken in traditionellem SQL können Sie mit *alias*`.`*column_name* oder *column_name* auf die Spalten verweisen.

JSON

- Dokument – Sie können auf JSON-Dokumentfelder als *alias*`.`*name* zugreifen. Auch der Zugriff auf verschachtelte Felder ist möglich, z. B *alias*`.`*name1*`.`*name2*`.`*name3*.
- Liste – Sie können über nullbasierte Indizes mit dem Operator `[]` auf Elemente in einer JSON-Liste zugreifen. Beispielsweise lässt sich das zweite Element einer Liste mithilfe von *alias*`[1]` aufrufen. Sie können den Zugriff auf Listenelemente mit dem Zugriff auf Felder kombinieren, z. B *alias*`.`*name1*`.`*name2*`[1].`*name3*.
- Beispiele: Betrachten Sie dieses JSON-Objekt als Beispieldatensatz:

```
{"name": "Susan Smith",
"org": "engineering",
"projects":
    [
     {"project_name":"project1", "completed":false},
     {"project_name":"project2", "completed":true}
    ]
}
```

Beispiel 1

Die folgende Abfrage gibt die folgenden Ergebnisse zurück:

```
Select s.name from S3Object s
```

```
{"name":"Susan Smith"}
```

Beispiel 2

Die folgende Abfrage gibt die folgenden Ergebnisse zurück:

```
Select s.projects[0].project_name from S3Object s
```

```
{"project_name":"project1"}
```

Groß-/Kleinschreibung bei Header- und Attributnamen

In Amazon S3 Select können Sie mithilfe von doppelten Anführungszeichen angeben, dass bei Spalten-Headern (für CSV-Objekte) und Attributen (für JSON-Objekte) die Groß-/Kleinschreibung beachtet werden muss. Ohne doppelte Anführungszeichen muss die Groß-/Kleinschreibung bei Objekt-Headern und -attributen nicht berücksichtigt werden. Im Falle einer Zweideutigkeit wird ein Fehler ausgegeben.

In den folgenden Beispielen handelt es sich entweder um 1) Amazon-S3-Objekte im CSV-Format mit spezifizierten Spalten-Headern und mit auf „`''Use''`“ festgelegtem `FileHeaderInfo`-Wert für die Abfrageanforderung oder um 2) Amazon-S3-Objekte im JSON-Format mit spezifizierten Attributen.

Beispiel 1: Das abzufragende Objekt hat den Header oder das Attribut „`NAME`“.

- Mit folgendem Ausdruck werden Werte von dem Objekt erfolgreich zurückgegeben. Da keine Anführungszeichen vorhanden sind, berücksichtigt die Abfrage die Groß- und Kleinschreibung nicht.

```
SELECT s.name from S3Object s
```

- Der folgende Ausdruck führt zum Fehler 400 „`MissingHeaderName`“. Da Anführungszeichen vorhanden sind, berücksichtigt die Abfrage die Groß- und Kleinschreibung.

```
SELECT s."name" from S3Object s
```

Beispiel 2: Das abzufragende Amazon S3-Objekt hat einen Header oder ein Attribut mit „`NAME`“ sowie einen anderen Header oder ein anderes Attribut mit „`name`“.

- Der folgende Ausdruck führt zum Fehler 400 „`AmbiguousFieldName`“. Da keine Anführungszeichen vorhanden sind, wird die Groß-/Kleinschreibung nicht beachtet, aber es gibt zwei Übereinstimmungen, weshalb der Fehler ausgegeben wird.

```
SELECT s.name from S3Object s
```

- Mit folgendem Ausdruck werden Werte von dem Objekt erfolgreich zurückgegeben. Da Anführungszeichen vorhanden sind, berücksichtigt die Abfrage die Groß- und Kleinschreibung, weshalb es keine Mehrdeutigkeit gibt.

```
SELECT s."NAME" from S3Object s
```

Verwenden von reservierten Schlüsselwörtern als benutzerdefinierte Begriffe

Amazon S3 Select verfügt über eine Reihe reservierter Schlüsselwörter, die zur Ausführung der SQL-Ausdrücke benötigt werden, mit denen Objektinhalte abgefragt werden. Zu den reservierten Schlüsselwörtern zählen Funktionsnamen, Datentypen, Operatoren etc. Gelegentlich könnten sich benutzerdefinierte Begriffe – wie z. B. Spalten-Header (bei CSV-Dateien) oder Attribute (bei JSON-Objekten) – mit einem reservierten Schlüsselwort decken. In diesem Fall geben Sie mithilfe von doppelten Anführungszeichen an, dass Sie absichtlich einen benutzerdefinierten Begriff verwenden, der mit einem reservierten Schlüsselwort übereinstimmt. Andernfalls wird ein 400-Parse-Fehler ausgegeben.

Eine vollständige Liste der reservierten Schlüsselwörter finden Sie unter Reservierte Schlüsselwörter.

Im folgenden Beispiel handelt es sich entweder um 1) ein Amazon-S3-Objekt im CSV-Format mit spezifizierten Spalten-Headern und mit auf „`"Use"`“ festgelegtem `FileHeaderInfo`-Wert für die Abfrageanforderung oder um 2) ein Amazon-S3-Objekt im JSON-Format mit spezifizierten Attributen.

Beispiel: Das abzufragende Objekt hat einen Header oder ein Attribut mit dem Namen „CAST“, der ein reserviertes Schlüsselwort ist.

- Mit folgendem Ausdruck werden Werte von dem Objekt erfolgreich zurückgegeben. Da die Abfrage Anführungszeichen enthält, verwendet S3 Select den benutzerdefinierten Header oder das benutzerdefinierte Attribut.

```
SELECT s."CAST" from S3Object s
```

- Der folgende Ausdruck führt zu einem Analysefehler 400. Da in der Abfrage keine Anführungszeichen verwendet werden, steht CAST im Konflikt mit einem reservierten Schlüsselwort.

```
SELECT s.CAST from S3Object s
```

Skalare Ausdrücke

Innerhalb der WHERE-Klausel und der SELECT-Liste können Sie skalare SQL-Ausdrücke verwenden. Dies sind Ausdrücke, die skalare Werte zurückgeben. Sie haben das folgende Format:

- ***literal***

  Ein SQL-Literal

- ***column_reference***

  Ein Verweis auf eine Spalte in der Form *column_name* oder *alias*.*column_name*.

- ***unary_op expression***

  In diesem Fall ist ***unary_op*** ein unärer SQL-Operator.

- ***expression binary_op expression***

  In diesem Fall ist ***binary_op*** ein binärer SQL-Operator.

- ***func_name***

  In diesem Fall ist ***func_name*** der Name der aufzurufenden skalaren Funktion.

- ***expression*** [NOT] BETWEEN ***expression*** AND ***expression***

- ***expression*** LIKE ***expression*** [ESCAPE ***expression***]

Datentypen

Amazon S3 Select unterstützt mehrere primitive Datentypen.

Datentypkonvertierungen

Allgemein sollte die Funktion `CAST` verwendet werden, wenn sie definiert ist. Falls `CAST` nicht definiert ist, werden alle Eingabedaten als Zeichenfolge betrachtet. In diesem Fall müssen Sie Ihre Eingabedaten bei Bedarf in die relevanten Datentypen umwandeln.

Weitere Informationen zur Funktion `CAST` finden Sie unter CAST.

Unterstützte Datentypen

Amazon S3 Select unterstützt die folgenden primitiven Datentypen.

Name	Beschreibung	Beispiele
`bool`	Ein boolescher Wert, entweder `TRUE` oder `FALSE`.	`FALSE`
`int`, `integer`	Eine 8-Byte-Ganzzahl im Bereich -9.223.372.036.854 .775.808 bis 9.223.372.036.854.775.807.	`100000`
`string`	Eine UTF-8-kodierte Zeichenfolge mit variabler Länge. Das Standardlimit ist 1 Zeichen. Das maximale Zeichenlimit beträgt 2.147.483.647.	`'xyz'`
`float`	Eine 8-Byte-Gleitkommazahl.	`CAST(0.456 AS FLOAT)`
`decimal`, `numeric`	Eine Zahl mit der Basis 10 mit einer maximalen Genauigkeit von 38 (d. h. maximale Anzahl signifikanter Ziffern) und einer Größe zwischen -2^{31} und $2^{31}-1$ (d. h. der Exponent der Basis 10).	`123.456`

Name	Beschreibung	Beispiele
	Note Amazon S3 Select ignoriert die Skalierung und Genauigkeit, wenn Sie beide gleichzeitig bereitste llen.	
`timestamp`	Zeitstempel stellen einen bestimmten Zeitpunkt dar, enthalten immer einen lokalen Versatz und können beliebig genau sein. Im Textformat befolgen Zeitstempel den W3C-Hinweis zu Datums- und Zeitformaten, müssen jedoch mit dem Buchstaben „T“ enden, wenn die Präzision der Zeitstempel nicht mindestens ein ganzer Tag ist. Sekundenbruchteile mit mindestens einer Stelle bis zu beliebig vielen Stellen sind zulässig. Der lokale Zeitversatz kann entweder als Versatz im Format Stunde:Minute von UTC-Zeit oder als Buchstabe „Z“ zur Angabe einer lokalen UTC-Zeit angegeben werden. Lokale Zeitversätze sind für Zeitstempel mit Uhrzeit erforderl ich, für reine Datumswerte jedoch nicht zulässig.	`CAST('2007-04-05T14:30Z' AS TIMESTAMP)`

Unterstützte Parquet-Typen

Amazon S3 Select unterstützt die folgenden Parquet-Typen.

- `DATE`
- `DECIMAL`
- `ENUM`
- `INT(8)`
- `INT(16)`
- `INT(32)`
- `INT(64)`
- `LIST`

> **Note**
>
> Für die Ausgabe vom Parquet-Typ `LIST` unterstützt Amazon S3 Select nur das JSON-Format. Wenn die Abfrage die Daten jedoch auf einfache Werte beschränkt, kann der Parquet-Typ `LIST` auch im CSV-Format abgefragt werden.

- `STRING`
- `TIMESTAMP`-unterstützte Präzision (`MILLIS`/`MICROS`/`NANOS`)

> **Note**
>
> Zeitstempel, die als `INT(96)` gespeichert werden, werden nicht unterstützt. Aufgrund des Bereichs des Typs `INT(64)` können Zeitstempel in der Einheit `NANOS` nur Werte zwischen `1677-09-21 00:12:43` und `2262-04-11 23:47:16` darstellen. Werte, die außerhalb dieses Bereichs liegen, können mit der `NANOS`-Einheit nicht dargestellt werden.

Zuordnung von Parquet-Typen zu unterstützten Datentypen in Amazon S3 Select

Parquet-Typen	Unterstützte Datentypen
`DATE`	`timestamp`
`DECIMAL`	`decimal`, `numeric`
`ENUM`	`string`
`INT(8)`	`int`, `integer`
`INT(16)`	`int`, `integer`
`INT(32)`	`int`, `integer`

Parquet-Typen	Unterstützte Datentypen
INT(64)	decimal, numeric
LIST	Jeder Parquet-Typ in der Liste wird auf den entsprechenden Datentyp abgebildet.
STRING	string
TIMESTAMP	timestamp

Operatoren

Amazon S3 Select unterstützt die folgenden Operatoren.

Logische Operatoren

- AND
- NOT
- OR

Vergleichsoperatoren

- <
- >
- <=
- >=
- =
- <>
- !=
- BETWEEN
- IN – Beispiel: IN ('a', 'b', 'c')

Mustervergleichsoperatoren

- `LIKE`
- `_` (Vergleicht ein beliebiges Zeichen)
- `%` (Vergleicht eine beliebige Zeichenreihenfolge)

Einheitliche Operatoren

- `IS NULL`
- `IS NOT NULL`

Mathematische Operatoren

Es werden Addition, Subtraktion, Multiplikation, Division und Modulo wie folgt unterstützt:

- `+`
- `-`
- `*`
- `/`
- `%`

Rangfolge der Operatoren

Die folgende Tabelle zeigt die Rangfolge der Operatoren in absteigender Reihenfolge.

Operator oder Element	Assoziativität	Erforderlich
`-`	rechts	unär minus
`*`, `/`, `%`	links	Multiplikation, Division und Modulo
`+`, `-`	links	Addition und Subtraktion

Operator oder Element	Assoziativität	Erforderlich
IN		Mitgliedschaft festlegen
BETWEEN		in Bereich enthalten
LIKE		Zeichenfolgenübereinstimmung
<>		kleiner als, größer als
=	rechts	Gleichheit, Zuweisung
NOT	rechts	logische Negation
AND	links	logische Verbindung
OR	links	logische Disjunktion

Reservierte Schlüsselwörter

Nachfolgend finden Sie eine Liste der reservierten Schlüsselwörter für Amazon S3 Select. Zu diesen Schlüsselwörtern zählen Funktionsnamen, Datentypen, Operatoren und so weiter, die zur Ausführung der SQL-Ausdrücke benötigt werden, mit denen Objektinhalte abgefragt werden.

```
absolute
action
add
all
allocate
```

```
alter
and
any
are
as
asc
assertion
at
authorization
avg
bag
begin
between
bit
bit_length
blob
bool
boolean
both
by
cascade
cascaded
case
cast
catalog
char
char_length
character
character_length
check
clob
close
coalesce
collate
collation
column
commit
connect
connection
constraint
constraints
continue
convert
corresponding
```

```
count
create
cross
current
current_date
current_time
current_timestamp
current_user
cursor
date
day
deallocate
dec
decimal
declare
default
deferrable
deferred
delete
desc
describe
descriptor
diagnostics
disconnect
distinct
domain
double
drop
else
end
end-exec
escape
except
exception
exec
execute
exists
external
extract
false
fetch
first
float
for
```

```
foreign
found
from
full
get
global
go
goto
grant
group
having
hour
identity
immediate
in
indicator
initially
inner
input
insensitive
insert
int
integer
intersect
interval
into
is
isolation
join
key
language
last
leading
left
level
like
limit
list
local
lower
match
max
min
minute
```

```
missing
module
month
names
national
natural
nchar
next
no
not
null
nullif
numeric
octet_length
of
on
only
open
option
or
order
outer
output
overlaps
pad
partial
pivot
position
precision
prepare
preserve
primary
prior
privileges
procedure
public
read
real
references
relative
restrict
revoke
right
rollback
```

```
rows
schema
scroll
second
section
select
session
session_user
set
sexp
size
smallint
some
space
sql
sqlcode
sqlerror
sqlstate
string
struct
substring
sum
symbol
system_user
table
temporary
then
time
timestamp
timezone_hour
timezone_minute
to
trailing
transaction
translate
translation
trim
true
tuple
union
unique
unknown
unpivot
update
```

```
upper
usage
user
using
value
values
varchar
varying
view
when
whenever
where
with
work
write
year
zone
```

SQL-Funktionen

Amazon S3 Select unterstützt die folgenden SQL-Funktionen.

Themen

- Aggregationsfunktionen
- Konditionale Funktionen
- Konvertierungs-Funktionen
- Datumsfunktionen
- Zeichenfolgenfunktionen

Aggregationsfunktionen

Amazon S3 Select unterstützt die folgenden aggregierten Funktionen.

Funktion	Argumenttyp	Rückgabetyp
`AVG(`*`expressio`* *`n`* `)`	`INT`, `FLOAT`, `DECIMAL`	`DECIMAL` für ein `INT`-Argument , `FLOAT` für

Funktion	Argumenttyp	Rückgabetyp
		ein Gleitkomma-Argument; andernfalls gleich dem Argumentdatentyp.
COUNT	-	INT
MAX(*expression*)	INT, DECIMAL	Entspricht dem Argumenttyp.
MIN(*expression*)	INT, DECIMAL	Entspricht dem Argumenttyp.
SUM(*expression*)	INT, FLOAT, DOUBLE, DECIMAL	INT für ein INT-Argument, FLOAT für ein Gleitkomma-Argument; andernfalls gleich dem Argumentdatentyp.

SUMBeispiel für

Verwenden Sie einen SUM-Ausdruck, um die gesamten Objektgrößen eines Ordners in einem S3-Bestandslistenbericht zusammenzufassen.

Der folgende S3-Bestandslistenbericht ist eine CSV-Datei, die mit GZIP komprimiert wurde. Sie hat drei Spalten.

- Die erste Spalte enthält den Namen des S3-Buckets (*DOC-EXAMPLE-BUCKET*), für den der S3-Bestandslistenbericht bestimmt ist.

- Die zweite Spalte enthält den Objektschlüsselnamen, der das Objekt eindeutig im Bucket identifiziert.

  Der *example-folder/*-Wert in der ersten Zeile steht für den Ordner *example-folder*. Wenn Sie in Amazon S3 einen Ordner in Ihrem Bucket anlegen, erstellt S3 ein 0-Byte-Objekt mit einem Schlüssel, der auf den von Ihnen angegebenen Ordnernamen festgelegt ist.

  Der *example-folder/object1*-Wert in der zweiten Zeile steht für das Objekt *object1* im Ordner *example-folder*.

  Der *example-folder/object2*-Wert in der dritten Zeile steht für das Objekt *object2* im Ordner *example-folder*.

  Weitere Informationen über S3-Ordner finden Sie unter Organisieren von Objekten in der Amazon S3-Konsole mithilfe von Ordnern.
- Die dritte Spalte enthält die Objektgröße in Byte.

```
"DOC-EXAMPLE-BUCKET","example-folder/","0"
"DOC-EXAMPLE-BUCKET","example-folder/object1","2011267"
"DOC-EXAMPLE-BUCKET","example-folder/object2","1570024"
```

Zur Nutzung eines SUM-Ausdrucks für die Berechnung der Gesamtgröße des Ordners *example-folder* führen Sie die SQL-Abfrage mit Amazon S3 Select aus.

```
SELECT SUM(CAST(_3 as INT)) FROM s3object s WHERE _2 LIKE 'example-folder/%' AND _2 !=
 'example-folder/';
```

Abfrageergebnis:

```
3581291
```

Konditionale Funktionen

Amazon S3 Select unterstützt die folgenden bedingten Funktionen.

Themen

- CASE
- COALESCE

- NULLIF

CASE

Der CASE-Ausdruck ist ein bedingter Ausdruck, der sich mit den `if/then/else`-Anweisungen anderer Sprachen vergleichen lässt. CASE wird verwendet, um ein Ergebnis anzugeben, wenn es mehrere Bedingungen gibt. Es gibt zwei Arten von CASE-Ausdrücken: einfach und gesucht.

In einfachen CASE-Ausdrücken wird ein Ausdruck mit einem Wert verglichen. Wenn keine Übereinstimmung gefunden wird, wird die in der THEN-Klausel angegebene Aktion angewendet. Wenn keine Übereinstimmung gefunden wird, wird die in der ELSE-Klausel angegebene Aktion angewendet.

In gesuchten CASE-Ausdrücken wird jeder CASE-Ausdruck auf der Basis eines booleschen Ausdrucks evaluiert und die CASE-Anweisung gibt den ersten übereinstimmenden CASE-Ausdruck zurück. Wenn in den WHEN-Klauseln kein übereinstimmender CASE-Ausdruck gefunden wird, wird die Aktion in der ELSE-Klausel zurückgegeben.

Syntax

> **Note**
>
> Derzeit unterstützt Amazon S3 Select weder `ORDER BY` noch Abfragen, die neue Zeilen enthalten. Achten Sie darauf, Abfragen ohne Zeilenumbrüche zu verwenden.

Die folgende Zeichenfolge ist eine einfache CASE-Aussage, die verwendet wird, um Bedingungen abzugleichen:

```
CASE expression WHEN value THEN result [WHEN...] [ELSE result] END
```

Die folgende Zeichenfolge ist eine gesuchte CASE-Anweisung, die verwendet wird, um jede Bedingung auszuwerten:

```
CASE WHEN boolean condition THEN result [WHEN ...] [ELSE result] END
```

Beispiele

> **Note**
>
> Wenn Sie die Amazon-S3-Konsole verwenden, um die folgenden Beispiele auszuführen, und Ihre CSV-Datei eine Kopfzeile enthält, wählen Sie Exclude the first line of CSV data (Die erste CSV-Datenzeile ausschließen) aus.

Beispiel 1: Verwenden Sie einen einfachen CASE-Ausdruck, um `New York City` in einer Abfrage durch `Big Apple` zu ersetzen. Alle anderen Städtenamen werden durch `other` ersetzt.

```
SELECT venuecity, CASE venuecity WHEN 'New York City' THEN 'Big Apple' ELSE 'other' END
 FROM S3Object;
```

Abfrageergebnis:

```
venuecity        |   case
-----------------+-----------
Los Angeles      | other
New York City    | Big Apple
San Francisco    | other
Baltimore        | other
...
```

Beispiel 2: Verwenden Sie einen gesuchten CASE-Ausdruck, um Gruppennummern basierend auf dem `pricepaid`-Wert für einzelne Ticketverkäufe zuzuweisen:

```
SELECT pricepaid, CASE WHEN CAST(pricepaid as FLOAT) < 10000 THEN 'group 1' WHEN
 CAST(pricepaid as FLOAT) > 10000 THEN 'group 2' ELSE 'group 3' END FROM S3Object;
```

Abfrageergebnis:

```
pricepaid |  case
----------+---------
12624.00 | group 2
10000.00 | group 3
10000.00 | group 3
9996.00 | group 1
```

```
9988.00 | group 1
...
```

COALESCE

`COALESCE` wertet die Argumente nacheinander aus und gibt den ersten unbekannten Wert zurück, das heißt, den ersten Wert, der nicht null oder nicht fehlend ist. Null- und fehlende Werte werden von der Funktion nicht übernommen.

Syntax

```
COALESCE ( expression, expression, ... )
```

Parameter

expression

Der Zielausdruck, der von der Funktion verwendet wird.

Beispiele

```
COALESCE(1)                -- 1
COALESCE(null)             -- null
COALESCE(null, null)       -- null
COALESCE(missing)          -- null
COALESCE(missing, missing) -- null
COALESCE(1, null)          -- 1
COALESCE(null, null, 1)    -- 1
COALESCE(null, 'string')   -- 'string'
COALESCE(missing, 1)       -- 1
```

NULLIF

Bei zwei Ausdrücken, die das gleiche Auswertungsergebnis haben, gibt `NULLIF` `NULL` zurück. Andernfalls gibt `NULLIF` das Auswertungsergebnis für den ersten Ausdruck zurück.

Syntax

```
NULLIF ( expression1, expression2 )
```

Parameter

expression1, *expression2*

Die Zielausdrücke, die von der Funktion verwendet werden.

Beispiele

```
NULLIF(1, 1)             -- null
NULLIF(1, 2)             -- 1
NULLIF(1.0, 1)           -- null
NULLIF(1, '1')           -- 1
NULLIF([1], [1])         -- null
NULLIF(1, NULL)          -- 1
NULLIF(NULL, 1)          -- null
NULLIF(null, null)       -- null
NULLIF(missing, null)    -- null
NULLIF(missing, missing) -- null
```

Konvertierungs-Funktionen

Amazon S3 Select unterstützt die folgende Konvertierungsfunktion.

Themen

- CAST

CAST

Die CAST-Funktion konvertiert ein Element, z. B. einen Ausdruck zur Auswertung eines einzelnen Werts, von einem Typ in einen anderen.

Syntax

```
CAST ( expression AS data_type )
```

Parameter

expression

Eine Kombination von Werten, Operatoren und SQL-Funktionen, die zu einem Wert ausgewertet werden können

data_type

Der Zieldatentyp, z. B. INT, in den der Ausdruck umgewandelt werden soll Eine Liste der unterstützten Datentypen finden Sie unter Datentypen.

Beispiele

```
CAST('2007-04-05T14:30Z' AS TIMESTAMP)
CAST(0.456 AS FLOAT)
```

Datumsfunktionen

Amazon S3 Select unterstützt die folgenden Datumsfunktionen.

Themen

- DATE_ADD
- DATE_DIFF
- EXTRACT
- TO_STRING
- TO_TIMESTAMP
- UTCNOW

DATE_ADD

Bei einem Datumsteil, einer Menge und einem Zeitstempel gibt DATE_ADD einen aktualisierten Zeitstempel zurück, indem der Datumsteil anhand der Menge modifiziert wird.

Syntax

```
DATE_ADD( date_part, quantity, timestamp )
```

Parameter

date_part

Gibt den zu modifizierenden Teil des Datums an. Dabei kann es sich um einen der folgenden Werte handeln:

- Jahr

- Monat
- Tag
- Stunde
- Minute
- Sekunde

quantity

Der Wert, der auf den aktualisierten Zeitstempel anzuwenden ist. Positive „*quantity*“-Werte werden zum „date_part“-Wert des Zeitstempels addiert, negative Werte werden subtrahiert.

timestamp

Der Zielzeitstempel, der von der Funktion verwendet wird.

Beispiele

```
DATE_ADD(year, 5, `2010-01-01T`)                -- 2015-01-01 (equivalent to
 2015-01-01T)
DATE_ADD(month, 1, `2010T`)                     -- 2010-02T (result will add precision
 as necessary)
DATE_ADD(month, 13, `2010T`)                    -- 2011-02T
DATE_ADD(day, -1, `2017-01-10T`)                -- 2017-01-09 (equivalent to
 2017-01-09T)
DATE_ADD(hour, 1, `2017T`)                      -- 2017-01-01T01:00-00:00
DATE_ADD(hour, 1, `2017-01-02T03:04Z`)          -- 2017-01-02T04:04Z
DATE_ADD(minute, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:05:05.006Z
DATE_ADD(second, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:04:06.006Z
```

DATE_DIFF

Bei einem Datumsteil und zwei gültigen Zeitstempeln gibt DATE_DIFF die Differenz in Datumsteilen an. Sofern der *date_part*-Wert von *timestamp1* größer ist als der *date_part*-Wert von *timestamp2*, wird eine negative Ganzzahl zurückgegeben. Wenn der *date_part*-Wert von *timestamp1* kleiner ist als der *date_part*-Wert von *timestamp2*, wird eine positive Ganzzahl zurückgegeben.

Syntax

```
DATE_DIFF( date_part, timestamp1, timestamp2 )
```

Parameter

date_part

Gibt den zu vergleichenden Teil der Zeitstempel an. Die Definition von `date_part` finden Sie unter DATE_ADD.

timestamp1

Der erste Zeitstempel für den Vergleich.

timestamp2

Der zweite Zeitstempel für den Vergleich.

Beispiele

```
DATE_DIFF(year, `2010-01-01T`, `2011-01-01T`)            -- 1
DATE_DIFF(year, `2010T`, `2010-05T`)                     -- 4 (2010T is equivalent to
 2010-01-01T00:00:00.000Z)
DATE_DIFF(month, `2010T`, `2011T`)                       -- 12
DATE_DIFF(month, `2011T`, `2010T`)                       -- -12
DATE_DIFF(day, `2010-01-01T23:00`, `2010-01-02T01:00`) -- 0 (need to be at least 24h
 apart to be 1 day apart)
```

EXTRACT

Bei einem Datumsteil und einem Zeitstempel gibt `EXTRACT` den Datumsteilwert des Zeitstempels zurück.

Syntax

```
EXTRACT( date_part FROM timestamp )
```

Parameter

date_part

Gibt den zu extrahierenden Teil der Zeitstempel an. Dabei kann es sich um einen der folgenden Werte handeln:

- `YEAR`
- `MONTH`

- DAY
- HOUR
- MINUTE
- SECOND
- TIMEZONE_HOUR
- TIMEZONE_MINUTE

timestamp

Der Zielzeitstempel, der von der Funktion verwendet wird.

Beispiele

```
EXTRACT(YEAR FROM `2010-01-01T`)                              -- 2010
EXTRACT(MONTH FROM `2010T`)                                   -- 1 (equivalent to
 2010-01-01T00:00:00.000Z)
EXTRACT(MONTH FROM `2010-10T`)                                -- 10
EXTRACT(HOUR FROM `2017-01-02T03:04:05+07:08`)                -- 3
EXTRACT(MINUTE FROM `2017-01-02T03:04:05+07:08`)              -- 4
EXTRACT(TIMEZONE_HOUR FROM `2017-01-02T03:04:05+07:08`)       -- 7
EXTRACT(TIMEZONE_MINUTE FROM `2017-01-02T03:04:05+07:08`)     -- 8
```

TO_STRING

Bei einem Zeitstempel und einem Formatmuster gibt TO_STRING eine Zeichenfolgendarstellung des Zeitstempels im angegebenen Format zurück.

Syntax

```
TO_STRING ( timestamp time_format_pattern )
```

Parameter

timestamp

Der Zielzeitstempel, der von der Funktion verwendet wird.

time_format_pattern

Eine Zeichenfolge mit folgenden speziellen Zeichenbedeutungen:

Format	Beispiel	Beschreibung
yy	69	Jahreszahl mit 2 Ziffern
y	1969	Jahreszahl mit 4 Ziffern
yyyy	1969	Jahreszahl mit 4 Ziffern, mit Nullen aufgefüllt
M	1	Monatsname
MM	01	Monatsnam e, mit Nullen aufgefüllt
MMM	Jan	Abkürzung des Monatsnam ens
MMMM	January	Vollständiger Monatsnamen
MMMMM	J	Erster Buchstabe des Monatsnam ens (HINWEIS: Dieses Format kann nicht mit der Funktion „TO_TIMEST AMP “ verwendet werden.)

Format	Beispiel	Beschreibung
d	2	Monatstag (1-31)
dd	02	Monatstag , mit Nullen aufgefüllt (01-31)
a	AM	AM oder PM
h	3	Stunde (1-12)
hh	03	Stunde, mit Nullen aufgefüllt (01-12)
H	3	Stunde (0-23)
HH	03	Stunde, mit Nullen aufgefüllt (00-23)
m	4	Minute (0-59)
mm	04	Minute, mit Nullen aufgefüllt (00-59)
s	5	Sekunde (0-59)

Format	Beispiel	Beschreibung
`ss`	`05`	Sekunde, mit Nullen aufgefüllt (00-59)
`S`	`0`	Sekundenbruchteil (Genauigkeit: 0,1, Bereich: 0,0-0,9)
`SS`	`6`	Sekundenbruchteil (Genauigkeit: 0,01, Bereich: 0,0-0,99)
`SSS`	`60`	Sekundenbruchteil (Genauigkeit: 0,001, Bereich: 0,0-0,999)
…	…	…
`SSSSSSSSS`	`60000000`	Sekundenbruchteil (max. Genauigkeit: 1 Nanosekunde, Bereich: 0,0-0,999999999)
`n`	`60000000`	Nanosekunde

Format	Beispiel	Beschreibung
X	+07 oder Z	Offset in Stunden oder „Z“ bei“ Offset = 0
XX oder XXXX	+0700 oder Z	Offset in Stunden und Minuten oder „Z“ bei Offset = 0
XXX oder XXXXX	+07:00 oder Z	Offset in Stunden und Minuten oder „Z“ bei Offset = 0
x	7	Offset in Stunden
xx oder xxxx	700	Offset in Stunden und Minuten
xxx oder xxxxx	+07:00	Offset in Stunden und Minuten

Beispiele

```
TO_STRING(`1969-07-20T20:18Z`,  'MMMM d, y')                    -- "July 20, 1969"
TO_STRING(`1969-07-20T20:18Z`, 'MMM d, yyyy')                   -- "Jul 20, 1969"
TO_STRING(`1969-07-20T20:18Z`, 'M-d-yy')                        -- "7-20-69"
TO_STRING(`1969-07-20T20:18Z`, 'MM-d-y')                        -- "07-20-1969"
TO_STRING(`1969-07-20T20:18Z`, 'MMMM d, y h:m a')               -- "July 20, 1969 8:18
 PM"
```

```
TO_STRING(`1969-07-20T20:18Z`, 'y-MM-dd''T''H:m:ssX')          --
 "1969-07-20T20:18:00Z"
TO_STRING(`1969-07-20T20:18+08:00Z`, 'y-MM-dd''T''H:m:ssX')    --
 "1969-07-20T20:18:00Z"
TO_STRING(`1969-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXX')  --
 "1969-07-20T20:18:00+0800"
TO_STRING(`1969-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXXX') --
 "1969-07-20T20:18:00+08:00"
```

TO_TIMESTAMP

Bei einer Zeichenfolge wandelt `TO_TIMESTAMP` diese in einen Zeitstempel um. `TO_TIMESTAMP` ist der umgekehrte Vorgang von `TO_STRING`.

Syntax

```
TO_TIMESTAMP ( string )
```

Parameter

`string`

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
TO_TIMESTAMP('2007T')                          -- `2007T`
TO_TIMESTAMP('2007-02-23T12:14:33.079-08:00') -- `2007-02-23T12:14:33.079-08:00`
```

UTCNOW

`UTCNOW` gibt die aktuelle Zeit in UTC als Zeitstempel zurück.

Syntax

```
UTCNOW()
```

Parameter

`UTCNOW` nutzt keine Parameter.

Beispiele

```
UTCNOW() -- 2017-10-13T16:02:11.123Z
```

Zeichenfolgenfunktionen

Amazon S3 Select unterstützt die folgenden String-Funktionen.

Themen

CHAR_LENGTH, CHARACTER_LENGTH

`CHAR_LENGTH` (oder `CHARACTER_LENGTH`) zählt die Anzahl der Zeichen in der angegebenen Zeichenfolge.

> **Note**
>
> `CHAR_LENGTH` und `CHARACTER_LENGTH` sind Synonyme.

Syntax

```
CHAR_LENGTH ( string )
```

Parameter

`string`

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
CHAR_LENGTH('')          -- 0
```

```
CHAR_LENGTH('abcdefg')   -- 7
```

LOWER

LOWER wandelt alle Großbuchstaben einer Zeichenfolge in Kleinbuchstaben um. Alle Zeichen, die keine Großbuchstaben sind, bleiben unverändert.

Syntax

```
LOWER ( string )
```

Parameter

string

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
LOWER('AbCdEfG!@#$') -- 'abcdefg!@#$'
```

SUBSTRING

SUBSTRING gibt bei einer Zeichenfolge, einem Startindex und (optional) einer Länge die Teilzeichenfolge vom Startindex bis zum Ende der Zeichenfolge oder bis zur angegebenen Länge zurück.

> **Note**
>
> Das erste Zeichen der Eingabezeichenfolge hat die Indexposition 1.
>
> - Wenn `start` < 1 ist und keine Länge angegeben wird, dann wird die Indexposition auf 1 festgelegt.
> - Wenn `start` < 1 ist und eine Länge angegeben wird, dann wird die Indexposition auf `start + length -1` festgelegt.
> - Wenn `start + length -1` < 0 ist, dann wird eine leere Zeichenfolge zurückgegeben.
> - Wenn `start + length -1` >= 0 ist, dann wird die Teilzeichenfolge beginnend bei Indexposition 1 mit der Länge `start + length - 1` zurückgegeben.

Syntax

```
SUBSTRING( string FROM start [ FOR length ] )
```

Parameter

string

Die Zielzeichenfolge, die von der Funktion verwendet wird.

start

Die Startposition der Zeichenfolge.

length

Die Länge der zurückzugebenden Teilzeichenfolge. Falls nicht angegeben, wird bis zum Ende der Zeichenfolge fortgefahren.

Beispiele

```
SUBSTRING("123456789", 0)      -- "123456789"
SUBSTRING("123456789", 1)      -- "123456789"
SUBSTRING("123456789", 2)      -- "23456789"
SUBSTRING("123456789", -4)     -- "123456789"
SUBSTRING("123456789", 0, 999) -- "123456789"
SUBSTRING("123456789", 1, 5)   -- "12345"
```

TRIM

Kürzt vorangestellte oder nachgestellte Zeichen aus einer Zeichenfolge. Standardmäßig wird ein Leerzeichen (`' '`) entfernt.

Syntax

```
TRIM ( [[LEADING | TRAILING | BOTH remove_chars] FROM] string )
```

Parameter

string

Die Zielzeichenfolge, die von der Funktion verwendet wird.

`LEADING | TRAILING | BOTH`

Dieser Parameter gibt an, ob vorangestellte oder nachgestellte Zeichen oder vorangestellte und nachgestellte Zeichen gekürzt werden sollen.

`remove_chars`

Die zu entfernenden Zeichen. *`remove_chars`* kann eine Zeichenfolge mit einer Länge von > 1 sein. Diese Funktion gibt die Zeichenfolge mit einem beliebigen Zeichen von *`remove_chars`* zurück, der am Anfang oder am Ende der entfernten Zeichenfolge stand.

Beispiele

```
TRIM('       foobar         ')              -- 'foobar'
TRIM('      \tfoobar\t         ')           -- '\tfoobar\t'
TRIM(LEADING FROM '       foobar         ')  -- 'foobar         '
TRIM(TRAILING FROM '       foobar         ') -- '       foobar'
TRIM(BOTH FROM '       foobar         ')     -- 'foobar'
TRIM(BOTH '12' FROM '1112211foobar22211122') -- 'foobar'
```

UPPER

UPPER wandelt alle Kleinbuchstaben einer Zeichenfolge in Großbuchstaben um. Alle Zeichen, die keine Kleinbuchstaben sind, bleiben unverändert.

Syntax

```
UPPER ( string )
```

Parameter

`string`

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
UPPER('AbCdEfG!@#$') -- 'ABCDEFG!@#$'
```

Ausführung umfangreicher Batch-Vorgänge für Amazon S3-Objekte durch.

Sie können S3-Batch-Vorgänge verwenden, um umfangreiche Batch-Vorgänge für Amazon S3-Objekte durchzuführen. S3-Batch-Vorgänge kann eine einzelne Operation für Listen von Amazon S3-Objekten durchführen, die Sie angeben. Ein einziger Auftrag kann eine festgelegte Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführen. Amazon S3 verfolgt den Fortschritt, versendet Benachrichtigungen und speichert einen detaillierten Abschlussbericht zu allen Aktionen. So profitieren Sie von einer vollständig verwalteten, prüfbaren und serverlosen Umgebung. Sie können S3-Batch-Vorgänge über die AWS Management Console, die AWS CLI, Amazon-SDKs oder die REST-API verwenden.

Verwenden Sie S3-BatchVorgänge, um Objekte zu kopieren und Objekt-Markierungen oder Zugriffskontrolllisten (ACLs) festzulegen. Sie können Objekt-Wiederherstellungen auch von S3 Glacier Flexible Retrieval aus initiieren oder eine AWS Lambda-Funktion aufrufen, um benutzerdefinierte Aktionen mit Ihren Objekten durchzuführen. Sie können diese Vorgänge für eine benutzerdefinierte Liste von Objekten durchführen oder mit einem Amazon S3 Inventory einfach Objektlisten erzeugen. Amazon S3-Batch-Vorgänge verwendet die gleichen Amazon S3-APIs, die Sie bereits mit Amazon S3 verwenden, sodass Ihnen die Benutzeroberfläche bereits vertraut ist.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets. Weitere Informationen zur Verwendung von Batch Operations mit S3 Express One Zone und Verzeichnis-Buckets finden Sie unter Verwenden von Batch Operations mit S3 Express One Zone.

Grundlagen von S3-BatchVorgänge

Sie können S3-Batch-Vorgänge verwenden, um umfangreiche Batch-Vorgänge für Amazon S3-Objekte durchzuführen. S3-Batch-Vorgänge kann eine einzelne Operation oder Aktion in Listen von Amazon S3-Objekten ausführen, die Sie angeben.

Terminologie

In diesem Abschnitt werden die Begriffe Aufträge, Vorgänge und Aufgaben erwähnt, ihre Bedeutung ist wie folgt zu verstehen:

Aufgabe

Ein Auftrag ist die grundlegende Arbeitseinheit für S3-BatchVorgänge. Eine Aufgabe enthält alle Informationen, die erforderlich sind, um die angegebene Operation für die im Manifest aufgeführten Objekte auszuführen. Sobald Sie diese Informationen bereitgestellt und den Beginn des Auftrags angefordert haben, führt die Aufgabe die Operation für alle Objekte im Manifest durch.

Operation

Die Operation stellt die Art der API-Aktion dar, z. B. das Kopieren von Objekten, die von dem Batchoperations-Auftrag ausgeführt werden soll. Jeder Auftrag führt einen einzelnen Typ von Operation für alle Objekte aus, die im Manifest angegeben sind.

Aufgabe

Eine Aufgabe ist die Ausführungseinheit für einen Auftrag. Eine Aufgabe steht für einen einzelnen Aufruf einer Amazon S3- oder AWS Lambda-API-Operation zur Durchführung der Auftragsoperation für ein einzelnes Objekt. Während der Lebensdauer eines Auftrag erstellt S3-Batch-Vorgänge eine Aufgabe für jedes Objekt, das im Manifest angegeben ist.

Funktionsweise eines S3-Batchoperations-Auftrags

Ein Auftrag ist die grundlegende Arbeitseinheit für S3-BatchVorgänge. Eine Aufgabe enthält alle Informationen, die erforderlich sind, um die angegebene Operation für eine Liste von Objekten auszuführen. Um einen Auftrag zu erstellen, übergeben Sie S3-Batch-Vorgänge eine Liste von Objekten und geben die Aktion an, die für diese Objekte ausgeführt werden soll.

Informationen zu den Vorgängen, die S3 Batch Operations unterstützt, finden Sie unter Von S3 Batch-Vorgänge unterstützte Vorgänge.

Ein Batch-Auftrag führt die festgelegte Operation für jedes Objekt aus, das im Manifest enthalten ist. Ein Manifest listet die Objekte auf, die ein Batch-Auftrag verarbeiten soll, und es wird als Objekt in einem Bucket gespeichert. Sie können einen kommagetrennten Wert (CSV)-formatierten Amazon S3 Inventory-Bericht als Manifest nutzen, was die Erstellung umfassender Listen von Objekten in einem

Bucket erleichtert. Ein Manifest können Sie auch in einem einfachen CSV-Format festlegen, mit dem Sie Batch-Vorgänge für eine benutzerdefinierte Liste an Objekten durchführen können, die in einem einzelnen Bucket enthalten sind.

Nachdem Sie einen Auftrag erstellt haben, verarbeitet Amazon S3 die Liste der Objekte in dem Manifest und führt die festgelegte Operation für jedes Objekt aus. Während ein Auftrag ausgeführt wird, können Sie den Fortschritt programmgesteuert oder über die Amazon S3-Konsole überwachen. Sie können einen Auftrag auch so konfigurieren, dass er nach Ende der Ausführung einen Abschlussbericht erzeugt. Der Abschlussbericht beschreibt die Ergebnisse jeder Aufgabe, die von dem Auftrag durchgeführt wurde. Weitere allgemeine Informationen zur Überwachung von Aufträgen finden Sie unter Verwalten von S3-Batch-Vorgangsaufträgen.

Tutorial zu S3-Batchvorgängen

Im folgenden Tutorial werden vollständige end-to-end Verfahren für einige Batchoperationenaufgaben vorgestellt.

- Tutorial: Batch-Transcodierung von Videos mit S3- AWS Lambda Batchoperationen und AWS Elemental MediaConvert

Erteilen von Berechtigungen für Amazon-S3-BatchVorgänge

Bevor Sie S3 Batch Operation-Aufträge erstellen und ausführen, müssen Sie die erforderlichen Berechtigungen erteilen. Um einen Amazon-S3-Batch-Vorgangsauftrag zu erstellen, ist die `s3:CreateJob`-Benutzerberechtigung erforderlich. Dieselbe Entität, die den Auftrag erstellt, muss auch über die `iam:PassRole` Berechtigung verfügen, die für den Auftrag angegebene AWS Identity and Access Management (IAM)-Rolle an Batch Operations zu übergeben.

Allgemeine Informationen zur Angabe von IAM-Ressourcen finden Sie in den IAM JSON-Richtlinienelements-Ressourcen im IAM-Benutzerhandbuch. Die folgenden Abschnitte enthalten Informationen zum Erstellen einer IAM-Rolle und zum Anhängen von Richtlinien.

Themen

- Erstellen einer IAM-Rolle für S3-BatchVorgänge
- Anfügen von Berechtigungsrichtlinien

Erstellen einer IAM-Rolle für S3-BatchVorgänge

Amazon S3 benötigt Berechtigungen, um S3-Batch-Vorgänge für Sie auszuführen. Sie erteilen diese Berechtigungen über eine AWS Identity and Access Management (IAM)-Rolle. Dieser Abschnitt enthält Beispiele zu den Vertrauens- und Berechtigungsrichtlinien, die Sie beim Erstellen einer IAM-Rolle verwenden. Weitere Informationen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch. Beispiele finden Sie unter Steuern von Berechtigungen für S3-Batch-Vorgänge mithilfe von Auftrags-Markierungen und Kopieren von Objekten mit S3 BatchVorgänge.

Sie können in Ihren IAM-Richtlinien auch Bedingungsschlüssel verwenden, um Zugriffsberechtigungen für S3-BatchVorgängeaufträge zu filtern. Weitere Informationen und eine vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Vertrauensrichtlinie

Sie fügen der Rolle die folgende Vertrauensrichtlinie an, um dem S3-BatchVorgängeaervice-Prinzipal zu erlauben, die IAM-Rolle zu übernehmen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

Anfügen von Berechtigungsrichtlinien

Je nach Art der Operation können Sie eine der folgenden Richtlinien anfügen.

Bevor Sie Berechtigungen konfigurieren, beachten Sie Folgendes:

- Unabhängig von der Operation benötigt Amazon S3 Berechtigungen, um das Manifestobjekt aus Ihrem S3-Bucket zu lesen und optional einen Bericht in Ihren Bucket zu schreiben. Daher enthalten alle der folgenden Richtlinien diese Berechtigungen.

- Für Manifeste von Amazon-S3-Bestandsbericht benötigt S3-Batch-Vorgänge die Berechtigung, das Manifest.json-Objekt und alle zugehörigen CSV-Datendateien zu lesen.
- Versionsspezifische Berechtigungen wie `s3:GetObjectVersion` sind nur erforderlich, wenn Sie die Versions-ID der Objekte festlegen.
- Wenn Sie S3-Batchoperationen für verschlüsselte Objekte ausführen, muss die IAM-Rolle auch Zugriff auf die AWS KMS Schlüssel haben, mit denen sie verschlüsselt wurden.
- Wenn Sie ein mit verschlüsseltes Bestandsberichtsmanifest einreichen AWS KMS, muss Ihre IAM-Richtlinie die Berechtigungen `"kms:Decrypt"` und `"kms:GenerateDataKey"` für das Objekt manifest.json und alle zugehörigen CSV-Datendateien enthalten.

Objekte kopieren: PutObject

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectTagging"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::DestinationBucket/*"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging",
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::SourceBucket",
                "arn:aws:s3:::SourceBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
            ]
        }
    ]
}
```

Objektmarkierung ersetzen: PutObjectTagging

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectTagging",
        "s3:PutObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::TargetResource/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::ManifestBucket/*"
      ]
    },
    {
```

```
      "Effect":"Allow",
      "Action":[
        "s3:PutObject"
      ],
      "Resource":[
        "arn:aws:s3:::ReportBucket/*"
      ]
    }
  ]
}
```

Objektmarkierung löschen: DeleteObjectTagging

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "s3:DeleteObjectTagging",
              "s3:DeleteObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3:::TargetResource/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
```

```
            ]
        }
    ]
}
```

Ersetzen Sie die Zugriffskontrollliste: PutObjectAcl

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectAcl",
        "s3:PutObjectVersionAcl"
      ],
      "Resource": "arn:aws:s3:::TargetResource/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::ManifestBucket/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject"
      ],
      "Resource":[
        "arn:aws:s3:::ReportBucket/*"
      ]
    }
  ]
}
```

Wiederherstellen von Objekten: RestoreObject

```
{
```

```
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
          "s3:RestoreObject"
      ],
      "Resource": "arn:aws:s3:::TargetResource/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::ManifestBucket/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject"
      ],
      "Resource":[
        "arn:aws:s3:::ReportBucket/*"
      ]
    }
  ]
}
```

Objektsperrenaufbewahrung anwenden: PutObjectRetention

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetBucketObjectLockConfiguration",
            "Resource": [
                "arn:aws:s3:::TargetResource"
            ]
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectRetention",
                "s3:BypassGovernanceRetention"
            ],
            "Resource": [
                "arn:aws:s3:::TargetResource/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
            ]
        }
    ]
}
```

Wenden Sie die rechtliche Aufbewahrungsfrist für die Objektsperre an: PutObjectLegalHold

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetBucketObjectLockConfiguration",
            "Resource": [
                "arn:aws:s3:::TargetResource"
            ]
```

```
        },
        {
            "Effect": "Allow",
            "Action": "s3:PutObjectLegalHold",
            "Resource": [
                "arn:aws:s3:::TargetResource/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
            ]
        }
    ]
}
```

Replizieren vorhandener Objekte: InitiateReplication mit einem von S3 generierten Manifest

Verwenden Sie diese Richtlinie, wenn Sie ein von S3 generiertes Manifest verwenden und speichern. Weitere Informationen zur Verwendung der Batchvorgänge zum Replizieren vorhandener Objekte finden Sie unter Replizieren bestehender Objekte mit S3-Batch-Replikation.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
```

```
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
      {
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:PutInventoryConfiguration"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***"
         ]
      },
      {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject"
         ],
         "Resource":[
            "arn:aws:s3:::*** completion report bucket ****/*",
            "arn:aws:s3:::*** manifest bucket ****/*"
         ]
      }
   ]
}
```

Replizieren vorhandener Objekte: InitiateReplication mit einem Benutzermanifest

Verwenden Sie diese Richtlinie, wenn Sie ein vom Benutzer bereitgestelltes Manifest verwenden. Weitere Informationen zur Verwendung der Batchvorgänge zum Replizieren vorhandener Objekte finden Sie unter Replizieren bestehender Objekte mit S3-Batch-Replikation.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
            {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject"
         ],
         "Resource":[
            "arn:aws:s3:::*** completion report bucket ****/*"
         ]
      }
   ]
}
```

Erstellen eines S3-Batch-Vorgangsauftrags

Mit S3 Batch Operations können Sie umfangreiche Stapelvorgänge für eine Liste bestimmter Amazon-S3-Objekte durchführen. In diesem Abschnitt werden die Informationen beschrieben, die Sie zum Erstellen eines S3-Batch-Vorgangsauftrags benötigen, sowie die Ergebnisse einer `CreateJob`-Anforderung. Sie finden dort auch Anweisungen zum Erstellen eines Batch-Operations-Auftrags mit der Amazon-S3-Konsole, AWS Command Line Interface (AWS CLI) und AWS SDK for Java.

Wenn Sie einen S3-Batchvorgangs-Auftrag erstellen, können Sie einen Abschlussbericht für alle oder nur für die fehlgeschlagenen Aufträge anfordern. Sofern mindestens eine Aufgabe erfolgreich aufgerufen wurde, generiert S3 Batch Operations einen Bericht für abgeschlossene, fehlgeschlagene oder abgebrochene Aufträge. Weitere Informationen finden Sie unter Beispiele: Abschlussberichte zu S3-BatchVorgänge.

Themen

- Batch-Vorgangsauftrag-Anforderungselemente
- Angeben eines Manifests

Batch-Vorgangsauftrag-Anforderungselemente

Um einen S3-Batch-Vorgangsauftrag zu erstellen, müssen Sie die folgenden Informationen angeben:

Operation

Geben Sie die Operation an, die S3-Batch-Vorgänge für die Objekte im Manifest ausführen soll. Jeder Operationstyp akzeptiert Parameter, die für diesen Vorgang spezifisch sind. Mit Batchoperationen können Sie eine Operation in großen Mengen ausführen, mit den gleichen Ergebnissen wie bei der Ausführung dieser Operation one-by-one für jedes Objekt.

Manifest

Das Manifest ist eine Liste aller Objekte, für die S3 Batch Operations die festgelegte Aktion ausführen soll. Sie können die folgenden Verfahren verwenden, um ein Manifest für einen Batch-Operations-Auftrag anzugeben:

- Erstellen Sie manuell Ihre eigene benutzerdefinierte Objektliste im CSV-Format.
- Wählen Sie einen vorhandenen Amazon S3 Inventory-Bericht im CSV-Format aus.
- Weisen Sie Batch Operations an, automatisch ein Manifest auf der Grundlage von Objektfilterkriterien zu generieren, die Sie bei der Erstellung Ihres Auftrags angeben. Diese Option ist für Stapelreplikationsaufträge verfügbar, die Sie in der Amazon-S3-Konsole erstellen, oder für jeden Auftragstyp, den Sie mithilfe der AWS CLI, der AWS-SDKs oder der Amazon-S3-REST-API erstellen.

> **Note**
>
> - Unabhängig davon, wie Sie Ihr Manifest angeben, muss die Liste selbst in einem Allzweck-Bucket gespeichert werden. Batch Operations kann keine vorhandenen

Manifeste aus Verzeichnis-Buckets importieren (oder generierte Manifeste in Verzeichnis-Buckets speichern). Im Manifest beschriebene Objekte können jedoch in Verzeichnis-Buckets gespeichert werden. Weitere Informationen finden Sie unter Verzeichnis-Buckets.

- Wenn sich die Objekte in Ihrem Manifest in einem versionsgesteuerten Bucket befinden, müssen Sie die Versions-IDs für die Objekte angeben, um den Vorgang für eine bestimmte Version auszuführen. Batch Operations führt den Vorgang für die neueste Version durch, wenn keine Versions-ID angegeben ist. Wenn Ihr Manifest ein Versions-ID-Feld enthält, müssen Sie eine Versions-ID für alle Objekte im Manifest angeben.

Weitere Informationen finden Sie unter Angeben eines Manifests.

Priorität

Mit Auftragsprioritäten können Sie angeben, welche relative Priorität dieser Auftrag gegenüber den anderen in Ihrem Konto ausgeführten Aufträgen besitzt. Eine höhere Nummer bedeutet eine höhere Priorität.

Auftragsprioritäten sind nur in Beziehung zu den für andere Aufträge in demselben Konto und in der derselben Region festgelegten Prioritäten bedeutsam. Sie können wählen, welches Nummerierungssystem für Sie funktioniert. So können Sie beispielsweise allen Wiederherstellen- (`RestoreObject`) Aufträgen die Priorität 1, allen Kopieren- (`CopyObject`) Aufträgen die Priorität 2 und allen Zugriffssteuerungslisten (ACLs) ersetzen (`PutObjectAcl`)-Aufträgen die Priorität 3 zuweisen.

S3 Batch Operations priorisiert Aufträge gemäß den Prioritätszahlen, garantiert aber keine strikte Sortierung. Daher sind Auftrags-Prioritäten nicht dazu geeignet, sicherzustellen, dass ein Auftrag vor einem anderen Auftrag gestartet oder beendet wird. Wenn Sie eine strikte Sortierung gewährleisten möchten, müssen Sie den Abschluss eines Auftrags abwarten, bevor Sie den nächsten starten.

RoleArn

Sie müssen eine AWS Identity and Access Management (IAM) Rolle angeben, um den Auftrag auszuführen. Die verwendete IAM-Rolle muss über ausreichende Berechtigungen verfügen, um die im Auftrag festgelegte Operation durchzuführen. Um z. B. einen `CopyObject`-Auftrag auszuführen, benötigt die IAM-Rolle die `s3:GetObject`-Berechtigung für den Quell-Bucket und die `s3:PutObject`-Berechtigung für den Ziel-Bucket. Außerdem braucht die Rolle die Berechtigungen, das Manifest zu lesen und den Auftragsabschlussbericht zu schreiben.

Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

Weitere Informationen zu Amazon-S3-Berechtigungen finden Sie unter Amazon S3-Richtlinienaktionen.

> **Note**
>
> Batch-Operations-Aufträge, die Aktionen für Verzeichnis-Buckets ausführen, erfordern bestimmte Berechtigungen. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

Bericht

Geben Sie an, ob S3-Batch-Vorgänge einen Abschlussbericht erstellen soll. Wenn Sie einen Auftragsabschlussbericht anfordern, müssen Sie auch die Parameter für den Bericht in diesem Element angeben. Die notwendigen Informationen umfassen:

- Den Bucket, in dem der Bericht gespeichert werden soll

  > **Note**
  >
  > Der Bericht muss in einem Allzweck-Bucket gespeichert werden. Batch Operations kann Berichte nicht in Verzeichnis-Buckets speichern. Weitere Informationen finden Sie unter Verzeichnis-Buckets.

- Das Format des Berichts
- Ob der Bericht die Details aller oder nur fehlgeschlagener Aufgaben enthält
- Eine optionale Präfix-Zeichenfolge

Markierungen (optional)

Sie können den Zugriff auf Ihre S3-Batch-Vorgänge-Aufträge kennzeichnen und steuern, indem Sie Markierungen hinzufügen. Sie können Tags verwenden, um zu identifizieren, wer für einen Batch-Operations-Auftrag verantwortlich ist, oder um zu steuern, wie Benutzer mit einzelnen Batch-Operations-Aufträgen interagieren. Das Vorhandensein von Aufgabe-Markierungen kann einem Benutzer die Möglichkeit gewähren oder einschränken, eine Aufgabe abzubrechen, eine Aufgabe im Bestätigungsstatus zu aktivieren oder die Prioritätsstufe einer Aufgabe zu ändern. Beispielsweise können Sie einem Benutzer die Berechtigung zum Aufrufen des `CreateJob`-

Vorgangs erteilen, vorausgesetzt, der Auftrag wird mit dem Tag `"Department=Finance"` erstellt.

Sie können Aufträge mit bereits zugeordneten Markierungen erstellen und Sie können bereits erstellten Aufträgen nachträglich Markierungen hinzufügen.

Weitere Informationen finden Sie unter the section called “Verwenden von Markierungen”.

Description (optional)

Um Ihren Auftrag zu verfolgen und zu überwachen, können Sie auch eine Beschreibung von bis zu 256 Zeichen angeben. Amazon S3 enthält diese Beschreibung immer dann, wenn Informationen über einen Auftrag zurückgegeben oder Auftragsdetails auf der Amazon-S3-Konsole angezeigt werden. So können Sie Aufträge ganz einfach nach den ihnen zugewiesenen Beschreibungen sortieren und filtern. Beschreibungen müssen nicht eindeutig sein, sodass Sie Beschreibungen auch als Kategorien (z. B. "Wöchentliche Aufträge zum Kopieren von Protokollen") verwenden können, um Gruppen ähnlicher Aufträge nachzuverfolgen.

Angeben eines Manifests

Ein Manifest ist ein Amazon-S3-Objekt, das Objektschlüssel enthält, die Amazon S3 bearbeiten soll. Sie können ein Manifest mit einer der folgenden Methoden bereitstellen:

- Erstellen Sie eine neue Manifestdatei manuell.
- Verwenden Sie ein vorhandenes Manifest.
- Weisen Sie Batch Operations an, automatisch ein Manifest auf der Grundlage von Objektfilterkriterien zu generieren, die Sie bei der Erstellung Ihres Auftrags angeben. Diese Option ist für Stapelreplikationsaufträge verfügbar, die Sie in der Amazon-S3-Konsole erstellen, oder für jeden Auftragstyp, den Sie mithilfe der AWS CLI, der AWS-SDKs oder der Amazon-S3-REST-API erstellen.

> **Note**
>
> Unabhängig davon, wie Sie Ihr Manifest angeben, muss die Liste selbst in einem Allzweck-Bucket gespeichert werden. Batch Operations kann keine vorhandenen Manifeste aus Verzeichnis-Buckets importieren (oder generierte Manifeste in Verzeichnis-Buckets speichern). Im Manifest beschriebene Objekte können jedoch in Verzeichnis-Buckets gespeichert werden. Weitere Informationen finden Sie unter Verzeichnis-Buckets.

Erstellen einer Manifestdatei

Um ein Manifest manuell zu erstellen, geben Sie den Objektschlüssel des Manifests, das ETag (Entity-Tag) und die optionale Versions-ID in einer CSV-formatierten Liste an. Die Inhalte des Manifests müssen per URL verschlüsselt sein.

Standardmäßig wendet Amazon S3 automatisch die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) an, um ein Manifest zu verschlüsseln, das in einen Amazon-S3-Bucket hochgeladen wird. Manifeste, die die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden, werden nicht unterstützt. Manifeste, die serverseitige Verschlüsselung mit AWS Key Management Service (SSE-KMS) AWS KMS-Schlüsseln (SSE-KMS) verwenden, werden nur unterstützt, wenn Sie Bestandsberichte im CSV-Format verwenden.

Ihr Manifest muss den Bucket-Namen, den Objektschlüssel und (optional) die Objektversion enthalten. Sämtliche anderen Felder in dem Manifest werden nicht von S3-Batch-Vorgänge verwendet.

> **Note**
>
> Wenn sich die Objekte in Ihrem Manifest in einem versionsgesteuerten Bucket befinden, müssen Sie die Versions-IDs für die Objekte angeben, um den Vorgang für eine bestimmte Version auszuführen. Batch Operations führt den Vorgang für die neueste Version durch, wenn keine Versions-ID angegeben ist. Wenn Ihr Manifest ein Versions-ID-Feld enthält, müssen Sie eine Versions-ID für alle Objekte im Manifest angeben.

Im Folgenden finden Sie ein Beispiel für ein Manifest im CSV-Format ohne Versions-IDs.

```
Examplebucket,objectkey1
Examplebucket,objectkey2
Examplebucket,objectkey3
Examplebucket,photos/jpgs/objectkey4
Examplebucket,photos/jpgs/newjersey/objectkey5
Examplebucket,object%20key%20with%20spaces
```

Im Folgenden finden Sie ein Beispiel für ein Manifest im CSV-Format mit Versions-IDs.

```
Examplebucket,objectkey1,PZ9ibn9D5lP6p298B7S9_ceqx1n5EJ0p
```

```
Examplebucket,objectkey2,YY_ouuAJByNW1LRBfFMfxMge7XQWxMBF
Examplebucket,objectkey3,jbo9_jhdPEyB4RrmOxWS0kU0EoNrU_oI
Examplebucket,photos/jpgs/objectkey4,6EqlikJJxLTsHsnbZbSRffn24_eh5Ny4
Examplebucket,photos/jpgs/newjersey/objectkey5,imHf3FAiRsvBW_EHB8GOu.NHunHO1gVs
Examplebucket,object%20key%20with%20spaces,9HkPvDaZY5MVbMhn6TMn1YTb5ArQAo3w
```

Angeben einer vorhandenen Manifestdatei

Sie können ein Manifest in einer Auftragserstellungsanforderung in einem der beiden folgenden Formate erstellen:

- Amazon-S3-Bestandsbericht – Muss ein CSV-formatierter Amazon-S3-Bestandsbericht sein. Sie müssen die `manifest.json`-Datei angeben, die mit dem Bestandsbericht verknüpft ist. Weitere Informationen zu Bestandsberichten finden Sie unter Amazon S3 Inventory. Enthält der Bestandsbericht Versions-IDs, führt S3-Batch-Vorgänge die Aktionen auf den spezifischen Objektversionen aus.

  > **Note**
  >
  > - S3 Batch Operations unterstützt Bestandsberichte im .CSV-Format, die mit SSE-KMS verschlüsselt sind.
  > - Wenn Sie ein Bestandsbericht-Manifest übermitteln, das mit SSE-KMS verschlüsselt ist, muss Ihre IAM-Richtlinie die Berechtigungen `"kms:Decrypt"` und `"kms:GenerateDataKey"` für das `manifest.json`-Objekt und alle zugehörigen CSV-Datendateien enthalten.

- CSV-Datei – Jede Zeile in der Datei muss den Bucket-Namen, den Objektschlüssel und optional die Objektversion enthalten. Objektschlüssel müssen URL-codiert sein, wie in den folgenden Beispielen gezeigt. Das Manifest muss Versions-IDs für alle Objekte enthalten oder auslassen. Weitere Informationen über das CSV-Manifest-Format finden Sie unter JobManifestSpec in der Amazon-Simple-Storage-Service-API-Referenz.

  > **Note**
  >
  > S3 Batch Operations unterstützt keine .CSV-formatierten Manifestdateien, die mit SSE-KMS verschlüsselt sind.

> **Important**
>
> Wenn Sie ein manuell erstelltes Manifest und einen versionsgesteuerten Bucket verwenden, wird empfohlen, die Versions-IDs für die Objekte anzugeben. Wenn Sie einen Auftrag erstellen, analysiert S3-Batch-Vorgänge das gesamte Manifest, bevor Sie den Auftrag ausführen. Es wird jedoch kein „Snapshot“ des Bucket-Zustands aufgenommen.
> Da Manifeste Milliarden von Objekten enthalten können, kann die Ausführung von Aufträgen längere Zeit dauern, was sich darauf auswirken kann, für welche Version eines Objekts der Auftrag durchgeführt wird. Angenommen, Sie überschreiben ein Objekt mit einer neuen Version, während ein Auftrag ausgeführt wird, und Sie haben keine Versions-ID für dieses Objekt angegeben. In diesem Fall führt Amazon S3 den Vorgang auf der neuesten Version des Objekts durch, und nicht auf der Version, die galt, als Sie den Auftrag erstellten. Die einzige Möglichkeit, dieses Verhalten zu verhindern, besteht darin, eine Versions-ID für die im Manifest enthaltenen Objekte anzugeben.

Automatisches Generieren eines Manifests

Sie können Amazon S3 Batch Operations anweisen, automatisch ein Manifest auf der Grundlage von Objektfilterkriterien zu generieren, die Sie bei der Erstellung Ihres Auftrags angeben. Diese Option ist für Stapelreplikationsaufträge verfügbar, die Sie in der Amazon-S3-Konsole erstellen, oder für jeden Auftragstyp, den Sie mithilfe der AWS CLI, der AWS-SDKs oder der Amazon-S3-REST-API erstellen. Weitere Informationen zur Batch-Replikationen finden Sie unter Replizieren bestehender Objekte mit S3-Batch-Replikation.

Um ein Manifest automatisch zu generieren, geben Sie im Rahmen Ihrer Anforderung zur Auftragserstellung die folgenden Elemente an:

- Informationen über den Bucket, der Ihre Quellobjekte enthält, einschließlich des Bucket-Eigentümers und des Amazon-Ressourcennamens (ARN)
- Informationen zur Manifestausgabe, einschließlich eines Flags zum Erstellen einer Manifestdatei, des Eigentümers des Ausgabe-Buckets, des ARN, des Präfixes, des Dateiformats und des Verschlüsselungstyps
- Optionale Kriterien zum Filtern von Objekten nach Erstellungsdatum, Schlüsselname, Größe, Speicherklasse und Tags

Objektfilterkriterien

Um die Liste der Objekte zu filtern, die in ein automatisch generiertes Manifest aufgenommen werden sollen, können Sie die folgenden Kriterien angeben. Weitere Informationen finden Sie unter JobManifestGeneratorFilter in der Amazon-S3-API-Referenz.

CreatedAfter

Falls angegeben, enthält das generierte Manifest nur Quell-Bucket-Objekte, die nach diesem Zeitpunkt erstellt wurden.

CreatedBefore

Falls angegeben, enthält das generierte Manifest nur Quell-Bucket-Objekte, die vor diesem Zeitpunkt erstellt wurden.

EligibleForReplication

Falls angegeben, enthält das generierte Manifest nur Objekte, wenn sie gemäß der Replikationskonfiguration im Quell-Bucket für die Replikation in Frage kommen.

KeyNameConstraint

Falls angegeben, enthält das generierte Manifest nur Quell-Bucket-Objekte, deren Objektschlüssel den für MatchAnySubstring, MatchAnyPrefixund angegebenen Zeichenfolgeneinschränkungen entsprechenMatchAnySuffix.

MatchAnySubstring – Falls angegeben, enthält das generierte Manifest Objekte, wenn die angegebene Zeichenfolge irgendwo in der Objektschlüsselzeichenfolge erscheint.

MatchAnyPrefix – Falls angegeben, enthält das generierte Manifest Objekte, wenn die angegebene Zeichenfolge am Anfang der Objektschlüsselzeichenfolge angezeigt wird.

MatchAnySuffix – Falls angegeben, enthält das generierte Manifest Objekte, wenn die angegebene Zeichenfolge am Ende der Objektschlüsselzeichenfolge angezeigt wird.

MatchAnyStorageClass

Falls angegeben, enthält das generierte Manifest nur Quell-Bucket-Objekte, die mit der angegebenen Speicherklasse gespeichert sind.

ObjectReplicationStatuses

Falls angegeben, enthält das generierte Manifest nur Quell-Bucket-Objekte, die einen der angegebenen Replikationsstatus haben.

ObjectSizeGreaterThanBytes

Falls angegeben, enthält das generierte Manifest nur Quell-Bucket-Objekte, deren Dateigröße die angegebene Byteanzahl überschreitet.

ObjectSizeLessThanBytes

Falls angegeben, enthält das generierte Manifest nur Quell-Bucket-Objekte, deren Dateigröße die angegebene Byteanzahl unterschreitet.

> **Note**
>
> Die meisten Aufträge mit automatisch generierten Manifesten können nicht geklont werden. Stapelreplikationsaufträge können geklont werden, es sei denn, sie verwenden die Manifest-Filterkriterien `KeyNameConstraint`, `MatchAnyStorageClass`, `ObjectSizeGreaterThanBytes` oder `ObjectSizeLessThanBytes`.

Die Syntax für die Angabe von Manifestkriterien hängt von der Methode ab, mit der Sie Ihren Job erstellen. Beispiele finden Sie unter Erstellen eines-Auftrags.

Erstellen eines-Auftrags

Sie können Aufträge für S3 Batch Operations mithilfe der Amazon-S3-Konsole, AWS CLI-, AWS-SDKs oder der Amazon-S3-REST-API erstellen.

Weitere Informationen zum Erstellen einer Auftragsanfrage finden Sie unter Batch-Vorgangsauftrag-Anforderungselemente.

Voraussetzungen

Bevor Sie einen Batch-Operations-Auftrag erstellen, bestätigen Sie, dass Sie die entsprechenden Berechtigungen konfiguriert haben. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für Amazon-S3-BatchVorgänge.

Verwenden der S3-Konsole

So erstellen Sie einen Batch-Auftrag:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Klicken Sie im Navigationsbereich der Amazon-S3-Konsole auf Batch-Vorgänge (Batch-Vorgänge).
3. Wählen Sie Create job (Auftrag erstellen) aus.
4. Wählen Sie die Region aus, in der Sie Ihren Auftrag erstellen möchten.
5. Wählen Sie unter Manifest format (Manifestformat) das zu verwendende Manifestobjekt aus.

   - Wenn Sie die Option S3 inventory report (S3-Bestandsbericht) auswählen, müssen Sie den Pfad zu dem manifest.json-Objekt eingeben, das von Amazon S3 als Bestandteil des Bestandsberichts im CSV-Format generiert wurde. Darüber hinaus können Sie wahlweise auch die Versions-ID des Manifestobjekts eingeben, wenn Sie eine andere Version als die aktuelle verwenden möchten.
   - Wenn Sie CSV auswählen, geben Sie den Pfad zu einem CSV-formatierten Manifestobjekt ein. Das Manifestobjekt muss das in der Konsole beschriebene Format befolgen. Sie können zudem die Versions-ID des Manifestobjekts angeben, wenn Sie eine andere Version als die aktuelle verwenden möchten.

   > **Note**
   >
   > Die Amazon-S3-Konsole unterstützt die automatische Manifestgenerierung nur für Stapelreplikationsaufträge. Wenn Sie möchten, dass Amazon S3 für alle anderen Auftragstypen automatisch ein Manifest auf der Grundlage der von Ihnen angegebenen Filterkriterien generiert, müssen Sie Ihren Auftrag mithilfe der AWS CLI-, AWS-SDKs oder der Amazon-S3-REST-API konfigurieren.

6. Wählen Sie Next.
7. Wählen Sie unter Operation (Vorgang) den Vorgang aus, den Sie für alle Objekte in dem Manifest ausführen möchten. Geben Sie die Informationen für die ausgewählte Operation ein und wählen Sie anschließend Next (Weiter) aus.
8. Geben Sie die Informationen für Configure additional options (Zusätzliche Optionen konfigurieren) ein und wählen Sie anschließend Next (Weiter) aus.
9. Überprüfen Sie die Einstellungen unter Review (Überprüfen). Wenn Sie Änderungen vornehmen müssen, wählen Sie Previous. Wählen Sie andernfalls Create Job (Auftrag erstellen).

Verwenden des AWS CLI

Specify manifest

Das folgende Beispiel zeigt, wie Sie einen S3-Batch-Operations–`S3PutObjectTagging`Auftrag für Objekte erstellen, die in einer vorhandenen Manifestdatei aufgelistet sind.

So erstellen Sie den Batch-Vorgangsauftrag **`S3PutObjectTagging`**

1. Verwenden Sie die folgenden Befehle, um eine AWS Identity and Access Management (IAM)-Rolle zu erstellen und erstellen Sie anschließend eine IAM-Richtlinie, um die relevanten Berechtigungen zuzuweisen. Die folgende Rolle und die folgende Richtlinie gewähren Amazon S3 die Berechtigung zum Hinzufügen von Objekt-Tags, was Sie benötigen, wenn Sie in einem nachfolgenden Schritt den Auftrag erstellen.

   a. Verwenden Sie den folgenden Beispielbefehl, um eine IAM-Rolle für Batch Operations zu erstellen. Ersetzen Sie *`S3BatchJobRole`* durch den Namen, den Sie der Rolle geben möchten, um diesen Beispielbefehl zu verwenden.

```
aws iam create-role \
 --role-name S3BatchJobRole \
 --assume-role-policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}'
```

   Notieren Sie sich den Amazon-Ressourcenname (ARN) der Rolle. Sie benötigen den ARN zum Erstellen eines Auftrags.

   b. Verwenden Sie den folgenden Beispielbefehl zum Erstellen einer IAM-Richtlinie mit den erforderlichen Berechtigungen und fügen Sie sie der im vorherigen Schritt erstellten IAM-Rolle an. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter Erteilen von Berechtigungen für Amazon-S3-BatchVorgänge.

> **Note**
>
> Batch-Operations-Aufträge, die Aktionen für Verzeichnis-Buckets ausführen, erfordern bestimmte Berechtigungen. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

Um diesen Beispielbefehl zu verwenden, ersetzen Sie die *user input placeholders* wie folgt:

- Ersetzen Sie *S3BatchJobRole* durch den Namen Ihrer IAM-Rolle. Stellen Sie sicher, dass dieser Name mit dem Namen übereinstimmt, den Sie zuvor verwendet haben.
- Ersetzen Sie *PutObjectTaggingBatchJobPolicy* durch den Namen, den Sie Ihrer IAM-Richtlinie geben möchten.
- Ersetzen Sie *DOC-EXAMPLE-DESTINATION-BUCKET* durch den Namen des Buckets, der die Objekte enthält, auf die Sie Tags anwenden möchten.
- Ersetzen Sie *DOC-EXAMPLE-MANIFEST-BUCKET* durch den Namen des Buckets, der das Manifestenthält.
- Ersetzen Sie *DOC-EXAMPLE-REPORT-BUCKET* durch den Namen des Buckets, an den der Abschlussbericht übermittelt werden soll.

```
aws iam put-role-policy \
  --role-name S3BatchJobRole \
  --policy-name PutObjectTaggingBatchJobPolicy \
  --policy-document '{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectTagging",
        "s3:PutObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/*"
    },
    {
      "Effect": "Allow",
```

```
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-BUCKET/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject",
        "s3:GetBucketLocation"
      ],
      "Resource":[
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET/*"
      ]
    }
  ]
}'
```

2. Verwenden Sie den folgenden Beispielbefehl, um einen `S3PutObjectTagging`-Auftrag zu erstellen.

   Die `manifest.csv`-Datei stellt eine Liste mit Bucket- und Objektschlüsselwerten bereit. Der Auftrag wendet die angegebenen Tags auf Objekte an, die im Manifest aufgelistet sind. `ETag` ist das ETag des `manifest.csv`-Objekts, das Sie mit der Amazon-S3-Konsole abrufen können. Diese Anforderung enthält den `no-confirmation-required`-Parameter, damit Sie den Job ausführen können, ohne ihn mit dem `update-job-status`-Befehl bestätigen zu müssen. Weitere Informationen finden Sie unter create-job in der Referenz zum AWS CLI-Befehl.

   Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen. Ersetzen Sie *`IAM-role`* durch den ARN des Buckets, den Sie zuvor erstellt haben.

```
aws s3control create-job \
    --region us-west-2 \
    --account-id acct-id \
```

```
    --operation '{"S3PutObjectTagging": { "TagSet": [{"Key":"keyOne",
 "Value":"ValueOne"}] }}' \
    --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key"]},"Location":
{"ObjectArn":"arn:aws:s3:::my_manifests/
manifest.csv","ETag":"60e460c9d1046e73f7dde5043ac3ae85"}}' \
    --report '{"Bucket":"arn:aws:s3:::DOC-EXAMPLE-REPORT-
BUCKET","Prefix":"final-reports",
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}' \
    --priority 42 \
    --role-arn IAM-role \
    --client-request-token $(uuidgen) \
    --description "job description" \
    --no-confirmation-required
```

Als Antwort gibt Amazon S3 eine Auftrags-ID zurück (zum Beispie, `00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c`). Sie benötigen die Auftrags-ID, um den Auftrag zu identifizieren, zu überwachen und zu ändern.

Generate manifest

Das folgende Beispiel zeigt, wie Sie einen S3-Batch-Operations–`S3DeleteObjectTagging`-Auftrag erstellen, der automatisch ein Manifest auf der Grundlage Ihrer Objektfilterkriterien generiert. Zu diesen Kriterien gehören das Erstellungsdatum, der Schlüsselname, die Größe, die Speicherklasse und die Tags.

So erstellen Sie den Batch-Vorgangsauftrag **`S3DeleteObjectTagging`**

1. Verwenden Sie die folgenden Befehle, um eine AWS Identity and Access Management (IAM)-Rolle zu erstellen und erstellen Sie anschließend eine IAM-Richtlinie, um Berechtigungen zuzuweisen. Die folgende Rolle und die folgende Richtlinie gewähren Amazon S3 die Berechtigung zum Löschen von Objekt-Tags, was Sie benötigen, wenn Sie in einem nachfolgenden Schritt den Auftrag erstellen.

   a. Verwenden Sie den folgenden Beispielbefehl, um eine IAM-Rolle für Batch Operations zu erstellen. Ersetzen Sie *`S3BatchJobRole`* durch den Namen, den Sie der Rolle geben möchten, um diesen Beispielbefehl zu verwenden.

      ```
      aws iam create-role \
      ```

```
 --role-name S3BatchJobRole \
 --assume-role-policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}'
```

Notieren Sie sich den Amazon-Ressourcenname (ARN) der Rolle. Sie benötigen den ARN zum Erstellen eines Auftrags.

b. Verwenden Sie den folgenden Beispielbefehl zum Erstellen einer IAM-Richtlinie mit den erforderlichen Berechtigungen und fügen Sie sie der im vorherigen Schritt erstellten IAM-Rolle an. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter Erteilen von Berechtigungen für Amazon-S3-BatchVorgänge.

> **Note**
>
> Batch-Operations-Aufträge, die Aktionen für Verzeichnis-Buckets ausführen, erfordern bestimmte Berechtigungen. Weitere Informationen finden Sie unter AWS Identity and Access Management (IAM) für S3 Express One Zone.

Um diesen Beispielbefehl zu verwenden, ersetzen Sie die *`user input placeholders`* wie folgt:

- Ersetzen Sie *`S3BatchJobRole`* durch den Namen Ihrer IAM-Rolle. Stellen Sie sicher, dass dieser Name mit dem Namen übereinstimmt, den Sie zuvor verwendet haben.
- Ersetzen Sie *`DeleteObjectTaggingBatchJobPolicy`* durch den Namen, den Sie Ihrer IAM-Richtlinie geben möchten.
- Ersetzen Sie *`DOC-EXAMPLE-DESTINATION-BUCKET`* durch den Namen des Buckets, der die Objekte enthält, auf die Sie Tags anwenden möchten.

- Ersetzen Sie *DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET* durch den Namen des Buckets, in dem das Manifest gespeichert werden soll.
- Ersetzen Sie *DOC-EXAMPLE-REPORT-BUCKET* durch den Namen des Buckets, an den der Abschlussbericht übermittelt werden soll.

```
aws iam put-role-policy \
  --role-name S3BatchJobRole \
  --policy-name DeleteObjectTaggingBatchJobPolicy \
  --policy-document '{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:DeleteObjectTagging",
        "s3:DeleteObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/*"
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutInventoryConfiguration"
      ],
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
```

```
        "s3:PutObject",
        "s3:ListBucket"
      ],
      "Resource":[
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET/*",
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET/*"
      ]
    }
  ]
}'
```

2. Verwenden Sie den folgenden Beispielbefehl, um den `S3DeleteObjectTagging`-Auftrag zu erstellen.

   In diesem Beispiel geben die Werte im `--report`-Abschnitt den Bucket, das Präfix, das Format und den Umfang des Auftragsberichts an, der generiert wird. Der `--manifest-generator`-Abschnitt enthält Informationen über den Quell-Bucket, der die Objekte enthält, auf die der Auftrag wirkt, Informationen über die Manifest-Ausgabeliste, die für den Auftrag generiert wird, und Filterkriterien, um den Bereich der Objekte, die in das Manifest aufgenommen werden sollen, nach Erstellungsdatum, Namenseinschränkungen, Größe und Speicherklasse einzugrenzen. Der Befehl gibt auch die Priorität, die IAM-Rolle und die AWS-Region des Auftrags an.

   Weitere Informationen finden Sie unter create-job in der Referenz zum AWS CLI-Befehl.

   Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen. Ersetzen Sie *`IAM-role`* durch den ARN des Buckets, den Sie zuvor erstellt haben.

```
aws s3control create-job \
    --account-id 012345678901 \
    --operation '{
        "S3DeleteObjectTagging": {}
    }' \
    --report '{
        "Bucket":"arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET",
        "Prefix":"reports",
        "Format":"Report_CSV_20180820",
        "Enabled":true,
        "ReportScope":"AllTasks"
```

```
    }' \
    --manifest-generator '{
        "S3JobManifestGenerator": {
          "ExpectedBucketOwner": "012345678901",
          "SourceBucket": "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
          "EnableManifestOutput": true,
          "ManifestOutputLocation": {
            "ExpectedManifestBucketOwner": "012345678901",
            "Bucket": "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET",
            "ManifestPrefix": "prefix",
            "ManifestFormat": "S3InventoryReport_CSV_20211130"
          },
          "Filter": {
            "CreatedAfter": "2023-09-01",
            "CreatedBefore": "2023-10-01",
            "KeyNameConstraint": {
              "MatchAnyPrefix": [
                "prefix"
              ],
              "MatchAnySuffix": [
                "suffix"
              ]
            },
            "ObjectSizeGreaterThanBytes": 100,
            "ObjectSizeLessThanBytes": 200,
            "MatchAnyStorageClass": [
              "STANDARD",
              "STANDARD_IA"
            ]
          }
        }
      }' \
     --priority 2 \
     --role-arn IAM-role \
     --region us-east-1
```

Als Antwort gibt Amazon S3 eine Auftrags-ID zurück (zum Beispie, `00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c`). Sie benötigen diese Auftrags-ID, um den Auftrag zu identifizieren, zu überwachen oder zu ändern.

Verwendung der AWS SDK for Java

Specify manifest

Das folgende Beispiel zeigt, wie Sie einen S3-Batch-Operations–`S3PutObjectTagging`-Auftrag für Objekte erstellen, die in einer vorhandenen Manifestdatei aufgelistet sind. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

Example

```
package aws.example.s3control;



import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.*;

import java.util.UUID;
import java.util.ArrayList;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateJob {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String iamRoleArn = "IAM Role ARN";
        String reportBucketName = "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET";
        String uuid = UUID.randomUUID().toString();

        ArrayList tagSet = new ArrayList<S3Tag>();
        tagSet.add(new S3Tag().withKey("keyOne").withValue("ValueOne"));


        try {
            JobOperation jobOperation = new JobOperation()
                    .withS3PutObjectTagging(new S3SetObjectTaggingOperation()
                            .withTagSet(tagSet)
```

```
                );

            JobManifest manifest = new JobManifest()
                    .withSpec(new JobManifestSpec()
                            .withFormat("S3BatchOperations_CSV_20180820")
                            .withFields(new String[]{
                                    "Bucket", "Key"
                            }))
                    .withLocation(new JobManifestLocation()
                            .withObjectArn("arn:aws:s3:::my_manifests/manifest.csv")
                            .withETag("60e460c9d1046e73f7dde5043ac3ae85"));
            JobReport jobReport = new JobReport()
                    .withBucket(reportBucketName)
                    .withPrefix("reports")
                    .withFormat("Report_CSV_20180820")
                    .withEnabled(true)
                    .withReportScope("AllTasks");

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.createJob(new CreateJobRequest()
                    .withAccountId(accountId)
                    .withOperation(jobOperation)
                    .withManifest(manifest)
                    .withReport(jobReport)
                    .withPriority(42)
                    .withRoleArn(iamRoleArn)
                    .withClientRequestToken(uuid)
                    .withDescription("job description")
                    .withConfirmationRequired(false)
            );

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
```

```
    }
}
```

Generate manifest

Das folgende Beispiel zeigt, wie Sie einen S3-Batch-Operations–`s3PutObjectCopy`-Auftrag erstellen, der automatisch ein Manifest auf der Grundlage Ihrer Objektfilterkriterien, einschließlich des Erstellungsdatums, des Schlüsselnamens und der Größe, generiert. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

Example

```
        package aws.example.s3control;

    import com.amazonaws.AmazonServiceException;
    import com.amazonaws.SdkClientException;
    import com.amazonaws.auth.profile.ProfileCredentialsProvider;
    import com.amazonaws.services.s3control.AWSS3Control;
    import com.amazonaws.services.s3control.AWSS3ControlClient;
    import com.amazonaws.services.s3control.model.CreateJobRequest;
    import com.amazonaws.services.s3control.model.CreateJobResult;
    import com.amazonaws.services.s3control.model.JobManifestGenerator;
    import com.amazonaws.services.s3control.model.JobManifestGeneratorFilter;
    import com.amazonaws.services.s3control.model.JobOperation;
    import com.amazonaws.services.s3control.model.JobReport;
    import com.amazonaws.services.s3control.model.KeyNameConstraint;
    import com.amazonaws.services.s3control.model.S3JobManifestGenerator;
    import com.amazonaws.services.s3control.model.S3ManifestOutputLocation;
    import com.amazonaws.services.s3control.model.S3SetObjectTaggingOperation;
    import com.amazonaws.services.s3control.model.S3Tag;

    import java.time.Instant;
    import java.util.Date;
    import java.util.UUID;
    import java.util.ArrayList;

    import static com.amazonaws.regions.Regions.US_WEST_2;

    public class test {
        public static void main(String[] args) {
            String accountId = "012345678901";
            String iamRoleArn = "arn:aws:iam::012345678901:role/ROLE";
```

```
            String sourceBucketName = "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET";
            String reportBucketName = "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET";
            String manifestOutputBucketName = "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-
OUTPUT-BUCKET";
            String uuid = UUID.randomUUID().toString();
            long minimumObjectSize = 100L;

            ArrayList<S3Tag> tagSet = new ArrayList<>();
            tagSet.add(new S3Tag().withKey("keyOne").withValue("ValueOne"));

            ArrayList<String> prefixes = new ArrayList<>();
            prefixes.add("s3KeyStartsWith");

            try {
                JobOperation jobOperation = new JobOperation()
                        .withS3PutObjectTagging(new S3SetObjectTaggingOperation()
                                .withTagSet(tagSet)
                        );
                S3ManifestOutputLocation manifestOutputLocation = new
 S3ManifestOutputLocation()
                        .withBucket(manifestOutputBucketName)
                        .withManifestPrefix("manifests")
                        .withExpectedManifestBucketOwner(accountId)
                        .withManifestFormat("S3InventoryReport_CSV_20211130");

                JobManifestGeneratorFilter jobManifestGeneratorFilter = new
 JobManifestGeneratorFilter()
                        .withEligibleForReplication(true)
                        .withKeyNameConstraint(
                                new KeyNameConstraint()
                                        .withMatchAnyPrefix(prefixes))
                        .withCreatedBefore(Date.from(Instant.now()))
                        .withObjectSizeGreaterThanBytes(minimumObjectSize);

                S3JobManifestGenerator s3JobManifestGenerator = new
 S3JobManifestGenerator()
                        .withEnableManifestOutput(true)
                        .withManifestOutputLocation(manifestOutputLocation)
                        .withFilter(jobManifestGeneratorFilter)
                        .withSourceBucket(sourceBucketName);

                JobManifestGenerator jobManifestGenerator = new
 JobManifestGenerator()
                        .withS3JobManifestGenerator(s3JobManifestGenerator);
```

```
                JobReport jobReport = new JobReport()
                        .withBucket(reportBucketName)
                        .withPrefix("reports")
                        .withFormat("Report_CSV_20180820")
                        .withEnabled(true)
                        .withReportScope("AllTasks");

                AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                        .withCredentials(new ProfileCredentialsProvider())
                        .withRegion(US_WEST_2)
                        .build();

                CreateJobResult createJobResult = s3ControlClient.createJob(new
CreateJobRequest()
                        .withAccountId(accountId)
                        .withOperation(jobOperation)
                        .withManifestGenerator(jobManifestGenerator)
                        .withReport(jobReport)
                        .withPriority(42)
                        .withRoleArn(iamRoleArn)
                        .withClientRequestToken(uuid)
                        .withDescription("job description")
                        .withConfirmationRequired(true)
                );

                System.out.println("Created job " + createJobResult.getJobId());

            } catch (AmazonServiceException e) {
                // The call was transmitted successfully, but Amazon S3 couldn't
process
                // it and returned an error response.
                e.printStackTrace();
            } catch (SdkClientException e) {
                // Amazon S3 couldn't be contacted for a response, or the client
                // couldn't parse the response from Amazon S3.
                e.printStackTrace();
            }
        }
    }
```

Verwenden der REST-API

Sie können die REST-API verwenden, um einen Batchvorgangsauftrag zu erstellen. Weitere Informationen finden Sie unter CreateJob in der API-Referenz zu Amazon Simple Storage Service.

Auftragsantworten

Wenn die `CreateJob`-Anforderung erfolgreich ist, gibt Amazon S3 eine Auftrags-ID zurück. Die Auftrags-ID ist eine eindeutige Kennung, die Amazon S3 automatisch erstellt. Anhand dieser ID können Sie Ihren Batch-Vorgangsauftrag identifizieren und dessen Status überwachen.

Wenn Sie einen Auftrag über die AWS CLI, AWS-SDKs oder die REST-API erstellen, können Sie S3 Batch Operations so einrichten, dass es automatisch mit der Verarbeitung der Aufträge beginnt. Der Auftrag wird ausgeführt, sobald er bereit ist, und muss nicht hinter Aufträgen mit höherer Priorität warten.

Wenn Sie einen Auftrag über die Amazon-S3-Konsole erstellen, müssen Sie die Auftragsdetails überprüfen und bestätigen, dass Sie ihn ausführen möchten, bevor Batch Operations mit seiner Verarbeitung beginnen kann. Wenn ein Auftrag den Status „Ausgesetzt“ mehr als 30 Tage behält, schlägt er fehl.

Von S3 Batch-Vorgänge unterstützte Vorgänge

S3-Batch-Vorgänge unterstützt verschiedene Vorgänge. In den Themen in diesem Abschnitt wird jede dieser Vorgänge beschrieben.

Kopieren von Objekten

Die Operation Copy (Kopieren) kopiert jedes im Manifest angegebene Objekt. Sie können Objekte in einen Bucket in derselben AWS-Region oder in einen Bucket in einer anderen Region kopieren. S3-Batch-Vorgänge unterstützt die meisten über Amazon S3 verfügbaren Optionen zum Kopieren von Objekten. Zu diesen Optionen gehören das Festlegen von Objekt-Metadaten, das Festlegen von Berechtigungen sowie das Ändern der Speicherklasse eines Objekts.

Mit dem Kopiervorgang können Sie vorhandene nicht verschlüsselte Objekte kopieren und die neuen verschlüsselten Objekte in denselben Bucket schreiben. Weitere Informationen finden Sie unter Encrypting Objects with Amazon S3 Batch Operations (Verschlüsseln von Objekten mit Amazon S3 Batch Operations).

Wenn Sie Objekte kopieren, können Sie den Prüfsummenalgorithmus zur Berechnung der Prüfsumme des Objekts ändern. Wenn Objekte keine zusätzliche berechnete Prüfsumme

haben, können Sie eine hinzufügen, indem Sie den von Amazon S3 zu verwendenden Prüfsummenalgorithmus angeben. Weitere Informationen finden Sie unter Überprüfung der Objektintegrität.

Weitere Informationen zum Kopieren von Objekten in Amazon S3 sowie erforderliche und optionale Parameter finden Sie unter Objekte kopieren in diesem Handbuch und in CopyObject in der API-Referenz für Amazon Simple Storage Service.

Beschränkungen und Einschränkungen

- Alle Quellobjekte müssen sich in einem Bucket befinden.
- Alle Zielobjekte müssen sich in einem Bucket befinden.
- Sie benötigen Lese-Berechtigungen für den Quell-Bucket und Schreib-Berechtigungen für den Ziel-Bucket.
- Die maximale Größe eines Objekts, das kopiert werden soll, beträgt 5 GB.
- Wenn Sie versuchen, Objekte aus den Klassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive in die Speicherklasse S3 Standard zu kopieren, müssen Sie diese Objekte zuerst wiederherstellen. Weitere Informationen finden Sie unter Wiederherstellen eines archivierten Objekts.
- Copy-Aufgaben müssen in der Zielregion erstellt werden. Das ist die Region, in die Sie die Objekte kopieren möchten.
- Alle Copy-Optionen werden unterstützt, mit Ausnahme der bedingten Prüfungen von EMarkierungen und der serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C).
- Wenn die Buckets nicht versioniert sind, werden die Objekte mit dem gleichen Schlüsselnamen überschrieben.
- Objekte werden nicht zwingend in derselben Reihenfolge kopiert, in der sie im Manifest erscheinen. Wenn für versionierte Buckets die Beibehaltung der aktuellen/nicht aktuellen Versionsreihenfolge wichtig ist, sollten Sie zuerst alle nicht aktuellen Versionen kopieren. Kopieren Sie dann, nachdem der erste Auftrag abgeschlossen ist, die aktuellen Versionen in einen nachfolgenden Auftrag.
- Das Kopieren von Objekten zur Klasse Reduced Redundancy Storage (RRS) wird nicht unterstützt.

Kopieren von Objekten mit S3 BatchVorgänge

Sie können S3-Batch-Vorgänge verwenden, um einen PUT-Kopierauftrag zu erstellen, um Objekte innerhalb desselben Kontos oder in ein anderes Zielkonto zu kopieren. Die folgenden Abschnitte

enthalten Beispiele für die Speicherung und Verwendung eines Manifests, das sich in einem anderen Konto befindet. Im ersten Abschnitt können Sie Amazon S3 Inventory verwenden, um den Bestandsbericht für das Zielkonto bereitzustellen, damit er dort zur Auftragserstellung genutzt werden kann. Alternativ können Sie ein CSV-Manifest (Manifest mit Kommas als Trennzeichen) im Quell- oder Zielkonto verwenden, wie im zweiten Beispiel dargestellt. Das dritte Beispiel zeigt, wie der Kopiervorgang verwendet wird, um die S3-Bucket-Schlüsselverschlüsselung für vorhandene Objekte zu aktivieren.

Beispiele für Kopieroperation

- Verwenden eines im Zielkonto bereitgestellten Bestandsberichts, um Objekte über AWS-Konten hinweg zu kopieren
- Verwenden eines im Quellkonto gespeicherten CSV-Manifests, um Objekte über hinweg zu kopieren AWS-Konten
- Verwenden von S3-Batch-Vorgänge zum Verschlüsseln von Objekten mit S3-Bucket-Schlüssel

Verwenden eines im Zielkonto bereitgestellten Bestandsberichts, um Objekte über AWS-Konten hinweg zu kopieren

Verwenden Sie Amazon S3 Inventory, um einen Bestandsbericht zu erstellen, und erstellen Sie mithilfe des Berichts eine Liste von Objekten, die mit S3-Batch-Vorgänge kopiert werden sollen. Informationen zur Verwendung eines CSV-Manifests im Quell- oder Zielkonto finden Sie unter the section called “Verwenden eines CSV-Manifests, um Objekte über hinweg zu kopieren AWS-Konten”.

Der Amazon S3 Inventory generiert Bestandslisten der Objekte in einem Bucket. Die resultierende Liste wird in einer Ausgabedatei veröffentlicht. Der Bucket, für den die Bestandsliste erstellt wird, wird als Quell-Bucket bezeichnet, der Bucket, in dem die Bestandsberichtsdatei gespeichert wird, dagegen als Ziel-Bucket.

Der Amazon-S3-Inventory-Bericht kann so konfiguriert werden, dass er an ein anderes AWS-Konto übermittelt wird. S3-Batch-Vorgänge kann den Bestandsbericht dann lesen, wenn der Auftrag im Zielkonto erstellt wird.

Weitere Informationen zu Quell- und Ziel-Buckets für Amazon S3 Inventory finden Sie unter Quell- und Ziel-Buckets.

Die einfachste Möglichkeit, einen Bestand einzurichten, ist die Verwendung der AWS Management Console, Sie können aber auch REST-API, AWS Command Line Interface (AWS CLI) oder AWS-SDKs verwenden.

Die folgende Konsolenprozedur enthält die allgemeinen Schritte zum Einrichten von Berechtigungen für einen S3-Batchoperations-Auftrag. Bei diesem Verfahren kopieren Sie Objekte aus einem Quellkonto in ein Zielkonto, wobei der Bestandsbericht im Zielkonto gespeichert wird.

So richten Sie einen Amazon S3 Inventory für Quell- und Ziel-Buckets ein, die zu unterschiedlichen Konten gehören

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie einen Ziel-Bucket, in dem der Bestandsbericht gespeichert werden soll.

   Legen Sie einen Ziel-Manifest-Bucket fest, in dem der Bestandsbericht gespeichert werden soll. In diesem Verfahren ist das Zielkonto das Konto, zu dem der Ziel-Manifest-Bucket sowie der Bucket gehören, in den die Objekte kopiert werden.
3. Konfigurieren Sie eine Bestandsliste, um die Objekte in einem Quell-Bucket aufzulisten und die Liste in einem Ziel-Bucket zu veröffentlichen.

   Konfigurieren Sie eine Bestandsliste für einen Quell-Bucket. Wenn Sie dies tun, geben Sie den Ziel-Bucket an, in dem die Liste gespeichert werden soll. Der Bestandsbericht für den Quell-Bucket wird im Ziel-Bucket veröffentlicht. In diesem Verfahren ist das Quellkonto das Konto, zu dem der Quell-Bucket gehört.

   Informationen zur Verwendung der Konsole zum Konfigurieren einer Bestandsliste oder zum Verschlüsseln einer Bestandsdatei finden Sie unter Konfigurieren von Amazon S3 Inventory.

   Wählen Sie CSV als Ausgabeformat.

   Wenn Sie Informationen für den Ziel-Bucket eingeben, wählen Sie Buckets in another account (Buckets in einem anderen Konto). Geben Sie dann den Namen des Ziel-Manifest-Buckets ein. Optional können Sie die Konto-ID des Zielkontos eingeben.

   Nachdem die Bestands-Konfiguration gespeichert wurde, zeigt die Konsole eine Meldung wie die folgende an:

   Amazon S3 could not create a bucket policy on the destination bucket. Ask the destination bucket owner to add the following bucket policy to allow Amazon S3 to place data in that bucket. (Amazon S3 konnte keine Bucket-Richtlinie für den Ziel-Bucket erstellen.Bitten Sie den Eigentümer des Ziel-Buckets, die folgende Bucket-Richtlinie hinzuzufügen, damit Amazon S3 Daten in diesen Bucket einfügen kann.

Die Konsole zeigt dann eine Bucket-Richtlinie an, die Sie für den Ziel-Bucket verwenden können.

4. Kopieren Sie die in der Konsole angezeigte Ziel-Bucket-Richtlinie.
5. Fügen Sie die kopierte Bucket-Richtlinie im Zielkonto dem Ziel-Manifest-Bucket hinzu, in dem der Bestandsbericht gespeichert wird.
6. Erstellen Sie eine Rolle im Zielkonto, die auf der S3-BatchVorgängevertrauensrichtlinie basiert. Weitere Informationen zur Vertrauensrichtlinie finden Sie unter Vertrauensrichtlinie.

   Weitere Informationen zum Erstellen einer Rolle finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

   Geben Sie einen Namen für die Rolle ein (die Beispielrolle hat den Namen `BatchOperationsDestinationRoleCOPY`). Wählen Sie den Service S3 und dann den Anwendungsfall S3 Bucket Batch Operations (S3-Bucket-Batch-Vorgänge) aus, mit dem die Vertrauensrichtlinie der Rolle zugewiesen wird.

   Wählen Sie dann Create policy (Richtlinie erstellen) aus, um der Rolle die folgende Richtlinie anzufügen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsDestinationObjectCOPY",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectVersionAcl",
        "s3:PutObjectAcl",
        "s3:PutObjectVersionTagging",
        "s3:PutObjectTagging",
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": [
        "arn:aws:s3:::ObjectDestinationBucket/*",
```

```
        "arn:aws:s3:::ObjectSourceBucket/*",
        "arn:aws:s3:::ObjectDestinationManifestBucket/*"
      ]
    }
  ]
}
```

Die Rolle verwendet die Richtlinie, um die Berechtigung `batchoperations.s3.amazonaws.com` zum Lesen des Manifests im Ziel-Bucket zu gewähren. Zudem werden Berechtigungen für GET-Objekte, Zugriffskontrolllisten (ACLs), Markierungen und Versionen im Quell-Bucket der Objekte gewährt. Und es werden Berechtigungen für PUT-Objekte, ACLs, Markierungen sowie Versionen im Ziel-Bucket der Objekte gewährt.

7. Erstellen Sie im Quellkonto eine Bucket-Richtlinie für den Quell-Bucket, die der im vorherigen Schritt erstellten Rolle Berechtigungen für GET-Objekte, ACLs, Markierungen und Versionen im Quell-Bucket gewährt. Dieser Schritt ermöglicht es S3-BatchVorgänge, Objekte über die vertrauenswürdige Rolle aus dem Quell-Bucket abzurufen.

   Das folgende Beispiel zeigt eine Bucket-Richtlinie für das Quellkonto.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsSourceObjectCOPY",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
      },
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::ObjectSourceBucket/*"
    }
  ]
```

```
}
```

8. Sobald der Bestandsbericht verfügbar ist, erstellen Sie einen S3-Batchoperations-PUT-Auftrag zum Kopieren von Objekten im Zielkonto und wählen den Bestandsbericht im Ziel-Manifest-Bucket aus. Sie benötigen den ARN der im Zielkonto erstellten Rolle.

   Allgemeine Informationen zum Erstellen eines Auftrags finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags.

   Informationen zum Erstellen eines Auftrags mit der Konsole finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags.

Verwenden eines im Quellkonto gespeicherten CSV-Manifests, um Objekte über hinweg zu kopieren AWS-Konten

Sie können eine in einem anderen AWS-Konto gespeicherte CSV-Datei als Manifest für einen S3-Batchoperations-Auftrag verwenden. Informationen zur Verwendung eines S3-Inventory-Berichts finden Sie unter the section called “Verwenden eines Bestandsberichts, um Objekte über AWS-Konten hinweg zu kopieren”.

Das folgende Verfahren zeigt, wie Berechtigungen bei Verwendung eines S3-Batchoperations-Auftrags zum Kopieren von Objekten aus einem Quellkonto in ein Zielkonto unter Verwendung der im Quellkonto gespeicherten CSV-Manifestdatei eingerichtet werden.

So richten Sie ein in einem anderen gespeichertes CSV-Manifest ein AWS-Konto

1. Erstellen Sie eine Rolle im Zielkonto, die auf der S3-BatchVorgängevertrauensrichtlinie basiert. In diesem Verfahren ist das Zielkonto das Konto, in das die Objekte kopiert werden.

   Weitere Informationen zur Vertrauensrichtlinie finden Sie unter Vertrauensrichtlinie.

   Weitere Informationen zum Erstellen einer Rolle finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

   Wenn Sie die Rolle mit der Konsole erstellen, geben Sie einen Namen für die Rolle ein (der Name der Beispielrolle lautet `BatchOperationsDestinationRoleCOPY`). Wählen Sie den Service S3 und dann den Anwendungsfall S3 Bucket Batch Operations (S3-Bucket-Batch-Vorgänge) aus, mit dem die Vertrauensrichtlinie der Rolle zugewiesen wird.

Wählen Sie dann Create policy (Richtlinie erstellen) aus, um der Rolle die folgende Richtlinie anzufügen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsDestinationObjectCOPY",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectVersionAcl",
        "s3:PutObjectAcl",
        "s3:PutObjectVersionTagging",
        "s3:PutObjectTagging",
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": [
        "arn:aws:s3:::ObjectDestinationBucket/*",
        "arn:aws:s3:::ObjectSourceBucket/*",
        "arn:aws:s3:::ObjectSourceManifestBucket/*"
      ]
    }
  ]
}
```

Mit der Richtlinie gewährt die Rolle die Berechtigung `batchoperations.s3.amazonaws.com` zum Lesen des Manifests im Quell-Manifest-Bucket. Sie gewährt Berechtigungen für GET-Objekte, Zugriffskontrolllisten (ACLs), Markierungen und Versionen im Quell-Bucket der Objekte. Zudem gewährt sie Berechtigungen für PUT-Objekte, ACLs, Markierungen und Versionen im Ziel-Bucket der Objekte.

2. Erstellen Sie im Quellkonto eine Bucket-Richtlinie für den Bucket, die der im vorherigen Schritt erstellten Rolle Berechtigungen für GET-Objekte, ACLs, Markierungen und Versionen in Quell-Manifest-Bucket gewährt.

Dieser Schritt ermöglicht S3-Batch-Vorgänge das Lesen des Manifests unter Verwendung der vertrauenswürdigen Rolle. Weisen Sie die Bucket-Richtlinie dem Bucket zu, in dem sich das Manifest befindet.

Das folgende Beispiel zeigt eine Bucket-Richtlinie, die dem Quell-Manifest-Bucket zugewiesen werden soll.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsSourceManfiestRead",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::DestinationAccountNumber:user/ConsoleUserCreatingJob",
          "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
        ]
      },
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::ObjectSourceManifestBucket/*"
    }
  ]
}
```

Diese Richtlinie gewährt außerdem die erforderlichen Berechtigungen, damit ein Konsolenbenutzer, der einen Auftrag im Zielkonto erstellt, diese Berechtigungen über dieselbe Bucket-Richtlinie auch für den Ziel-Manifest-Bucket erhält.

3. Erstellen Sie im Quellkonto eine Bucket-Richtlinie für den Quell-Bucket, die der erstellten Rolle Berechtigungen für GET-Objekte, ACLs, Markierungen und Versionen im Quellobjekt-Bucket gewährt. S3-Batch-Vorgänge kann dann Objekte über die vertrauenswürdige Rolle aus dem Quell-Bucket abrufen.

   Es folgt ein Beispiel für die Bucket-Richtlinie für den Bucket, der die Quellobjekte enthält.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsSourceObjectCOPY",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
      },
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::ObjectSourceBucket/*"
    }
  ]
}
```

4. Erstellen Sie einen S3-Batchoperations-Auftrag im Zielkonto. Sie benötigen den Amazon-Ressourcennamen (ARN) der im Zielkonto erstellten Rolle.

   Allgemeine Informationen zum Erstellen eines Auftrags finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags.

   Informationen zum Erstellen eines Auftrags mit der Konsole finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags.

Verwenden von S3-Batch-Vorgänge zum Verschlüsseln von Objekten mit S3-Bucket-Schlüssel

In diesem Abschnitt verwenden Sie den Copy-Vorgang für Amazon S3-BatchVorgänge, um die Verschlüsselung von S3-Bucket-Schlüsseln für vorhandene Objekte zu identifizieren und zu aktivieren. Weitere Informationen zu S3-Bucket-Schlüsseln finden Sie unter Reduzieren des Preises von SSE-KMS mit Amazon-S3-Bucket-Schlüsseln und Konfigurieren des Buckets für die Verwendung eines S3-Bucket-Schlüssels mit SSE-KMS bei neuen Objekten.

In diesem Beispiel werden folgende Themen behandelt:

Themen

- Voraussetzungen
- Schritt 1: Abrufen der Liste von Objekten mithilfe von Amazon S3 Inventory
- Schritt 2: Filtern Sie Ihre Objektliste mit S3 Select
- Schritt 3: Einrichten und Ausführen des Auftrags für S3-BatchVorgänge
- Übersicht

Voraussetzungen

Zum Ausführen der Schritte in diesem Verfahren benötigen Sie ein AWS-Konto und mindestens einen S3-Bucket, um Ihre Arbeitsdateien und verschlüsselten Ergebnisse zu speichern. Möglicherweise finden Sie auch vieles aus der vorhandenen Dokumentation zu S3 Batch Operations nützlich, einschließlich der folgenden Themen:

- Grundlagen von S3-BatchVorgänge
- Erstellen eines S3-Batch-Vorgangsauftrags
- Von S3 Batch-Vorgänge unterstützte Vorgänge
- Verwalten von S3-Batch-Vorgangsaufträgen

Schritt 1: Abrufen der Liste von Objekten mithilfe von Amazon S3 Inventory

Geben Sie zunächst den S3-Bucket an, der die zu verschlüsselnden Objekte enthält, und rufen Sie eine Liste des Inhalts ab. Ein Amazon S3-Bestandsbericht ist die bequemste und kostengünstigste Methode, dies zu tun. Der Bericht enthält die Liste der Objekte in einem Bucket sowie die zugehörigen Metadaten. Der Quell-Bucket bezieht sich auf den im Bestand erfassten Bucket und der Ziel-Bucket bezieht sich auf den Bucket, in dem die Bestandsberichts-Datei gespeichert wird. Weitere Informationen zu Quell- und Ziel-Buckets für Amazon S3 Inventory finden Sie unter Amazon S3 Inventory.

Die einfachste Möglichkeit zum Einrichten eines Bestands ist mit der AWS Management Console. Sie können aber auch die REST-API, AWS Command Line Interface (AWS CLI), oder AWS-SDKs verwenden. Melden Sie sich bei der Konsole an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/. Wenn Fehler mit Berechtigung verweigert auftreten, fügen Sie eine Bucket-Richtlinie zu Ihrem Ziel-Bucket hinzu. Weitere Informationen finden Sie unter Gewähren von Berechtigungen für S3 Inventory und S3 Analytics.

Abrufen der Liste von Objekten mithilfe von S3 Inventory

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich die Option Buckets und wählen Sie einen Bucket aus, der Objekte enthält, die zu verschlüsseln sind.
3. Navigieren Sie im Tab Verwaltung zu dem Abschnitt Inventory configurationes (Bestands-Konfigurationen), und wählen Sie Create inventory configuration (Bestands-Konfiguration erstellen).
4. Geben Sie Ihrem neuen Bestand einen Namen, geben Sie den Namen des Ziel-S3-Buckets ein und erstellen Sie optional ein Zielpräfix für Amazon S3, um Objekte in diesem Bucket zuzuweisen.
5. Für das Ausgabeformat wählen Sie CSV.
6. (Optional) Wählen Sie im Abschnitt Zusätzliche Felder – optional die Option Verschlüsselung und alle anderen für Sie interessanten Listenfelder aus. Legen Sie die Häufigkeit für die Berichterstattung auf Daily (Täglich), damit der erste Bericht früher an Ihren Bucket geliefert wird.
7. Wählen Sie Save (Speichern) aus, um die Konfiguration zu speichern.

Amazon S3 benötigt bis zu 48 Stunden, den ersten Bericht bereitzustellen. Kehren Sie daher zurück, wenn der erste Bericht eingeht. Nachdem Sie Ihren ersten Bericht erhalten haben, fahren Sie mit dem nächsten Abschnitt fort, um den Inhalt Ihres S3-Bestandsberichts zu filtern. Wenn Sie keine Bestandsberichte für diesen Bucket mehr erhalten möchten, löschen Sie Ihre S3-Bestands-Konfiguration. Andernfalls liefert S3 Berichte nach einem täglichen oder wöchentlichen Zeitplan.

Eine Bestandsliste ist nicht eine einzelne point-in-time Ansicht aller Objekte. Bestandslisten sind fortlaufende Snapshots von Bucket-Elementen, die letztendlich konsistent sind (d. h. die Liste enthält möglicherweise keine vor kurzem hinzugefügten oder gelöschten Objekte). Die Kombination von S3 Inventory und S3-Batch-Vorgänge funktioniert am besten, wenn Sie mit statischen Objekten oder mit einem Objektsatz arbeiten, den Sie vor zwei oder mehr Tagen erstellt haben. Um mit neueren Daten zu arbeiten, verwenden Sie den API-Vorgang ListObjectsV2 (GET Bucket), um Ihre Objektliste manuell zu erstellen. Wiederholen Sie den Vorgang bei Bedarf für die nächsten Tage oder bis Ihr Bestandsbericht den gewünschten Status für alle Schlüssel anzeigt.

Schritt 2: Filtern Sie Ihre Objektliste mit S3 Select

Nachdem Sie Ihren S3-Bestandsbericht erhalten haben, können Sie den Inhalt des Berichts filtern, um nur die Objekte aufzulisten, die nicht mit S3-Bucket-Schlüsseln verschlüsselt sind. Wenn Sie möchten, dass alle Objekte Ihres Buckets mit S3-Bucket-Schlüsseln verschlüsselt werden, können

Sie diesen Schritt ignorieren. Durch das Filtern Ihres S3-Bestandsberichts in diesem Stadium sparen Sie jedoch Zeit und Kosten für das erneute Verschlüsseln von Objekten, die Sie zuvor verschlüsselt haben.

In den folgenden Schritten wird das Filtern mit Amazon S3 Select gezeigt. Sie können aber auchAmazon Athena verwenden. Um zu entscheiden, welches Tool verwendet werden soll, schauen Sie sich die `manifest.json`-Datei Ihres S3-Bestandsberichts an. Diese Datei listet die Anzahl der Datendateien auf, die diesem Bericht zugeordnet sind. Wenn die Zahl groß ist, verwenden Sie Amazon Athena, da es über mehrere S3-Objekte läuft, während S3 Select für jeweils ein Objekt funktioniert. Weitere Informationen zur gemeinsamen Verwendung von Amazon S3 und Athena finden Sie unter Abfragen von Amazon S3 Inventory mit Amazon Athena und Using Athena (Verwendung von Athena)Im Blogbeitrag Encrypting objects with Amazon S3 Batch Operations.

So filtern Sie Ihren S3-Bestandsbericht mithilfe von S3 Select

1. Öffnen Sie`manifest.json`-Datei aus Ihrem Bestandsbericht und schauen Sie sich den `fileSchema`-Abschnitt des JSON an. Dadurch wird die Abfrage informiert, die Sie für die Daten ausführen.

   Der folgende JSON ist eine Beispiels-`manifest.json`-Datei für einen CSV-formatierten Bestand in einem Bucket mit aktivierter Versionierung. Je nachdem, wie Sie Ihren Bestandsbericht konfiguriert haben, sieht Ihr Manifest möglicherweise anders aus.

   ```
     {
       "sourceBucket": "batchoperationsdemo",
       "destinationBucket": "arn:aws:s3:::testbucket",
       "version": "2021-05-22",
       "creationTimestamp": "1558656000000",
       "fileFormat": "CSV",
       "fileSchema": "Bucket, Key, VersionId, IsLatest, IsDeleteMarker,
    BucketKeyStatus",
       "files": [
         {
           "key": "demoinv/batchoperationsdemo/DemoInventory/data/009a40e4-
   f053-4c16-8c75-6100f8892202.csv.gz",
           "size": 72691,
           "MD5checksum": "c24c831717a099f0ebe4a9d1c5d3935c"
         }
       ]
     }
   ```

Wenn die Versionierung für den Bucket nicht aktiviert ist oder wenn Sie den Bericht für die neuesten Versionen ausführen möchten, ist die `fileSchema` `Bucket`, `Key`, und `BucketKeyStatus`.

Wenn die Versionsverwaltung aktiviert ist, beinhaltet das `fileSchema` je nach der Einrichtung des Bestandsberichts möglicherweise Folgendes: `Bucket`, `Key`, `VersionId`, `IsLatest`, `IsDeleteMarker`, `BucketKeyStatus`. Achten Sie also auf die Spalten 1, 2, 3 und 6, wenn Sie Ihre Abfrage ausführen.

S3 Batch Operations benötigt den Bucket, den Schlüssel und die Versions-ID als Eingabe, um den Auftrag auszuführen, zusätzlich zu dem Feld, nach dem gesucht werden soll, nämlich `BucketKeyStatus`. Das Versions-ID-Feld wird nicht benötigt. Diese Angabe ist jedoch hilfreich, wenn Sie mit einem versionierten Bucket arbeiten. Weitere Informationen finden Sie unter Arbeiten mit Objekten in einem versioning-fähigen Bucket.

2. Suchen Sie die Datendateien für den Bestandsbericht. Das `manifest.json`-Objekt listet die Datendateien unter files (Dateien) auf.
3. Klicken Sie nach dem Suchen und Auswählen der Datendatei in der S3-Konsole auf Actions (Aktionen) und wählen Sie dann Query with S3 Select (Abfragen mit S3 Select).
4. Bewahren Sie die Voreinstellung CSV, Komma, und GZIP ausgewählt, und wählen Sie Next (Weiter).
5. Um das Format Ihres Bestands zu überprüfen, bevor Sie fortfahren, wählen Sie Show file preview (Dateivorschau anzeigen).
6. Geben Sie die Spalten, auf die Sie verweisen möchten, in das SQL-Ausdruck-Feld ein und wählen Sie Run SQL (SQL ausführen) aus. Der folgende Ausdruck gibt die Spalten 1 bis 3 für alle Objekte zurück, die keinen S3-Bucket-Schlüssel konfiguriert haben.

```
select s._1, s._2, s._3 from s3object s where s._6 = 'DISABLED'
```

Nachfolgend sehen Sie einige Beispielergebnisse.

```
batchoperationsdemo,0100059%7Ethumb.jpg,lsrtIxksLu0R0ZkYPL.LhgD5caTYn6vu
batchoperationsdemo,0100074%7Ethumb.jpg,sd2M60g6Fdazoi6D5kNARIE7KzUibmHR
batchoperationsdemo,0100075%7Ethumb.jpg,TLYESLnl1mXD5c4BwiOIinqFrktddkoL
batchoperationsdemo,0200147%7Ethumb.jpg,amufzfMi_fEw0Rs99rxR_HrDFlE.l3Y0
batchoperationsdemo,0301420%7Ethumb.jpg,9qGU2SEscL.C.c_sK89trmXYIwooABSh
batchoperationsdemo,0401524%7Ethumb.jpg,ORnEWNuB1QhHrrYAGFsZhbyvEYJ3DUor
```

```
      batchoperationsdemo,200907200065HQ
%7Ethumb.jpg,d8LgvIVjbDR5mUVwW6pu9ahTfReyn5V4
      batchoperationsdemo,200907200076HQ
%7Ethumb.jpg,XUT25d7.gK40u_GmnupdaZg3BVx2jN40
      batchoperationsdemo,201103190002HQ
%7Ethumb.jpg,z.2sVRh0myqVi0BuIrngWlsRPQdb7qOS
```

7. Laden Sie die Ergebnisse herunter, speichern Sie sie in einem CSV-Format und laden Sie sie als Liste der Objekte für den S3-Batchoperations-Auftrag in Amazon S3 hoch.
8. Wenn Sie mehrere Manifestdateien haben, führen Sie auch auf diesen Query with S3 Select (Abfragen mit S3 Select) aus. Abhängig von der Größe der Ergebnisse können Sie die Listen kombinieren und einen einzelnen Auftrag für S3-Batch-Vorgänge ausführen oder jede Liste als separater Auftrag ausführen.

   Betrachten Sie den Preis für jede Ausführung eines S3-Batchoperation-Auftrags, wenn Sie die Anzahl der auszuführenden Aufträge festlegen.

Schritt 3: Einrichten und Ausführen des Auftrags für S3-BatchVorgänge

Nachdem Sie nun Ihre gefilterten CSV-Listen von S3-Objekten haben, können Sie mit dem S3-Batchoperations-Auftrag beginnen, um die Objekte mit S3-Bucket-Schlüssel zu verschlüsseln.

Ein Auftrag bezieht sich kollektiv auf die Liste (Manifeste) der bereitgestellten Objekte, die durchgeführte Operation und die angegebenen Parameter. Die einfachste Möglichkeit zum Verschlüsseln dieses Objektsatzes besteht in der Verwendung der PUT-Copy-Operation mit der gleichen Zielpräfix wie die im Manifest aufgeführten Objekte. Dadurch werden entweder die vorhandenen Objekte in einem nicht versionierten Bucket überschrieben, oder bei aktivierter Versionierung wird eine neuere, verschlüsselte Version der Objekte erstellt.

Geben Sie beim Kopieren der Objekte an, dass Amazon S3 das Objekt mit SSE-KMS-Verschlüsselung und S3 verschlüsseln soll. Dieser Auftrag kopiert die Objekte, sodass alle Objekte nach Fertigstellung ein aktualisiertes Erstellungsdatum anzeigen, unabhängig davon, wann Sie sie ursprünglich zu S3 hinzugefügt haben. Geben Sie auch die anderen Eigenschaften für Ihre Gruppe von Objekten als Teil des S3 Batchoperations-Auftrags an, einschließlich Objekt-Markierungen und Speicherklasse.

Teilschritte

- Einrichten Ihrer IAM-Richtlinie
- IAM-Rolle für Batch-Vorgänge einrichten

- Aktivieren von S3-Bucket-Schlüssel für einen vorhandenen Bucket
- Erstellen eines Auftrags für S3-BatchVorgänge
- Ausführen eines Auftrags für S3-BatchVorgänge
- Wissenswertes

Einrichten Ihrer IAM-Richtlinie

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Policy (Richtlinie) und dann Create Policy (Richtlinie erstellen).
3. Wählen Sie den Tab JSON. Klicken Sie auf Edit policy (Richtlinie bearbeiten) und fügen Sie die Beispielrichtlinie für IAM hinzu, die im folgenden Codeblock angezeigt wird.

   Nachdem Sie das Richtlinienbeispiel in Ihre IAM-Konsole kopiert haben, ersetzen Sie Folgendes:

   a. Ersetzen Sie *SOURCE_BUCKET_FOR_COPY* durch den Namen Ihres Quell-Buckets.
   b. Ersetzen Sie *DESTINATION_BUCKET_FOR_COPY* durch den Namen Ihres Ziel-Buckets.
   c. Ersetzen Sie *MANIFEST_KEY* durch den Namen Ihres Manifest-Objekts.
   d. Ersetzen Sie *REPORT_BUCKET* durch den Namen des Buckets, in dem die Berichte gespeichert werden sollen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CopyObjectsToEncrypt",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectTagging",
        "s3:PutObjectAcl",
        "s3:PutObjectVersionTagging",
        "s3:PutObjectVersionAcl",
        "s3:GetObject",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersion",
        "s3:GetObjectVersionAcl",
```

```
        "s3:GetObjectVersionTagging"
      ],
      "Resource": [
        "arn:aws:s3:::SOURCE_BUCKET_FOR_COPY/*",
        "arn:aws:s3:::DESTINATION_BUCKET_FOR_COPY/*"
      ]
    },
    {
      "Sid": "ReadManifest",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::MANIFEST_KEY"
    },
    {
      "Sid": "WriteReport",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::REPORT_BUCKET/*"
    }
  ]
}
```

4. Wählen Sie Next: Markierungen (Weiter: Markierungen) aus.
5. Wählen Sie alle gewünschten Markierungen aus (optional) und wählen Sie Next: Review (Weiter: Überprüfung).
6. Geben Sie einen Namen und optional eine Beschreibung für die Richtlinie an und wählen Sie dann Create policy (Richtlinie erstellen) aus.
7. Wählen Sie Review policy (Richtlinie überprüfen) aus und klicken Sie anschließend auf Save changes (Änderungen speichern).
8. Wenn Ihre S3-Batchoperations-Richtlinie jetzt abgeschlossen ist, bringt die Konsole Sie zur Seite der IAM-Richtlinien zurück. Markieren Sie den Richtliniennamen, klicken Sie auf die Schaltfläche links neben dem Richtliniennamen und wählen Sie Policy actions (Richtlinienaktionen), und dann Attach (Hinzufügen).

Um die neu erstellte Richtlinie einer IAM-Rolle zuzuordnen, wählen Sie die entsprechenden Benutzer, Gruppen oder Rollen in Ihrem Konto aus und wählen Sie Attach policy (Richtlinie anfügen) aus. Das bringt Sie zurück zur IAM-Konsole.

IAM-Rolle für Batch-Vorgänge einrichten

1. Wählen Sie im Navigationsbereich der IAM-Konsole Rollen und dann Rolle erstellen aus.
2. Wählen Sie AWS-Service, S3 und S3 Batch Operations. Wählen Sie dann Next: Permissions aus.
3. Beginnen Sie mit der Eingabe des Namens der IAM-Richtlinie, die Sie soeben erstellt haben. Aktivieren Sie das Kontrollkästchen nach dem Richtliniennamen, wenn es angezeigt wird, und wählen Sie Next: Markierungen (Weiter: Markierungen).
4. (Optional) Fügen Sie Markierungen hinzu, oder lassen Sie die Schlüssel- und Wertefelder für diese Übung leer. Klicken Sie auf Next: Review (Weiter: Prüfen).
5. Geben Sie einen Rollennamen ein, akzeptieren Sie die Standard-Beschreibung oder fügen Sie eine eigene hinzu. Wählen Sie Create role aus.
6. Stellen Sie sicher, dass der Benutzer, der den Auftrag erstellt, über die Berechtigungen im folgenden Beispiel verfügt.

   Ersetzen Sie *{ACCOUNT-ID}* durch Ihre AWS-Konto-ID und *{IAM_ROLE_NAME}* durch den Namen, den Sie auf die IAM-Rolle anwenden möchten, die Sie später während der Auftragserstellung in Batch Operations erstellen werden. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für Amazon-S3-BatchVorgänge.

```
{
"Sid": "AddIamPermissions",
"Effect": "Allow",
"Action": [
"iam:GetRole",
"iam:PassRole"
],
"Resource": "arn:aws:iam::ACCOUNT-ID:role/IAM_ROLE_NAME"
}
```

Aktivieren von S3-Bucket-Schlüssel für einen vorhandenen Bucket

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Bucket aus, für den Sie einen S3-Bucket-Schlüssel aktivieren möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Default encryption (Standard-Verschlüsselung) Edit (Bearbeiten) aus.
5. Unter Verschlüsselungstyp können Sie zwischen Von Amazon S3 verwalteten Schlüsseln (SSE-S3) und AWS Key Management Service-Schlüsseln (SSE-KMS) wählen.
6. Wenn Sie AWS Key Management Service-Schlüssel (SSE-KMS) ausgewählt haben, können Sie unter AWS KMS key mithilfe einer der folgenden Optionen den AWS KMS-Schlüssel angeben.

   - Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS-Schlüsseln wählen aus. Wählen Sie in der Liste verfügbarer Schlüssel einen KMS-Schlüssel mit symmetrischer Verschlüsselung aus, der sich in derselben Region wie Ihr Bucket befindet. Sowohl der von AWS verwaltete Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt.
   - Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS-Schlüssel-ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
   - Wählen Sie zum Erstellen eines neuen vom Kunden verwalteten Schlüssels in der AWS KMS-Konsole Erstellen eines KMS-Schlüssels aus.
7. Unter Bucket Key (Bucket-Schlüssel), wählen Sie Enable (Aktivieren) und dann Save changes (Änderungen speichern) aus.

Nachdem nun ein S3-Bucket-Schlüssel auf Bucket-Ebene aktiviert ist, übernehmen Objekte, die in diesen Bucket hochgeladen, geändert oder kopiert werden, diese Verschlüsselungskonfiguration standardmäßig. Dies umfasst Objekte, die mit Amazon S3 Batch Operations kopiert wurden.

Erstellen eines Auftrags für S3-BatchVorgänge

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich die Option Batch Operations (Batch-Vorgänge) und dann Create Job (Auftrag erstellen).
3. Wählen Sie die Region, in der Sie Ihre Objekte speichern und wählen Sie CSV als Manifest-Typ.

4. Geben Sie den Pfad ein, oder navigieren Sie zu der CSV-Manifestdatei, die Sie zuvor aus Ergebnisse von S3 Select (oder Athena) erstellt haben. Wenn Ihr Manifest Versions-IDs enthält, aktivieren Sie dieses Kontrollkästchen. Wählen Sie Next.
5. Wählen Sie die Copy-Operation und wählen Sie den Ziel-Bucket für den Kopiervorgang aus. Sie können serverseitige Verschlüsselung deaktiviert lassen. Solange S3 Bucket-Schlüssel für den Ziel-Bucket aktiviert ist, wendet der Kopiervorgang S3 Bucket-Schlüssel auf den Ziel-Bucket an.
6. (Optional) Wählen Sie nach Bedarf eine Speicherklasse und die anderen Parameter aus. Die in diesem Schritt angegebenen Parameter gelten für alle Vorgänge, die an den im Manifest aufgeführten Objekten ausgeführt werden. Wählen Sie Weiter aus.
7. Gehen Sie zum Konfigurieren der serverseitigen Verschlüsselung wie folgt vor:

   a. Wählen Sie unter Serverseitige Verschlüsselung eine der folgenden Optionen aus:

      - Um die Bucket-Einstellungen für die serverseitige Standardverschlüsselung von Objekten beizubehalten, wenn sie in Amazon S3 gespeichert werden, wählen Sie Keinen Verschlüsselungsschlüssel angeben aus. Solange S3-Bucket-Schlüssel für den Ziel-Bucket aktiviert sind, wendet der Kopiervorgang S3-Bucket-Schlüssel auf den Ziel-Bucket an.

        > **Note**
        >
        > Wenn die Bucket-Richtlinie für das angegebene Ziel vorschreibt, dass Objekte verschlüsselt werden müssen, bevor sie in Amazon S3 gespeichert werden, müssen Sie einen Verschlüsselungsschlüssel angeben. Andernfalls schlägt das Kopieren von Objekten in das Ziel fehl.

      - Um Objekte zu verschlüsseln, bevor sie in Amazon S3 gespeichert werden, wählen Sie Verschlüsselungsschlüssel angeben aus.

   b. Wenn Sie unter Verschlüsselungseinstellungen die Option Verschlüsselungsschlüssel angeben auswählen, müssen Sie entweder Verwenden von Ziel-Bucket-Einstellungen für die Standardverschlüsselung oder Überschreiben der Ziel-Bucket-Einstellungen für die Standardverschlüsselung auswählen.

   c. Wenn Sie Überschreiben der Ziel-Bucket-Einstellungen für die Standardverschlüsselung auswählen, müssen Sie die folgenden Verschlüsselungseinstellungen konfigurieren.

      i. Unter Verschlüsselungstyp müssen Sie entweder Von Amazon S3 verwaltete Schlüssel (SSE-S3) oder AWS Key Management Service-Schlüssel (SSE-KMS) auswählen.

SSE-S3 verwendet für die Verschlüsselung der einzelnen Objekte eine der stärksten Blockverschlüsselungen: 256-bit Advanced Encryption Standard (AES-256). Mit SSE-KMS erhalten Sie mehr Kontrolle über Ihren Schlüssel. Weitere Informationen finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) und Verwenden der serverseitigen Verschlüsselung mit - AWS KMS Schlüsseln (SSE-KMS).

ii. Wenn Sie AWS Key Management Service-Schlüssel (SSE-KMS) auswählen, können Sie unter AWS KMS key mithilfe einer der folgenden Optionen Ihren AWS KMS key angeben.

- Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS keys wählen und anschließend einen KMS-Schlüssel mit symmetrischer Verschlüsselung aus, der sich in derselben Region wie Ihr Bucket befindet. Sowohl der von AWS verwaltete Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt.
- Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS-Schlüssel-ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
- Wählen Sie zum Erstellen eines neuen vom Kunden verwalteten Schlüssels in der AWS KMS-Konsole Erstellen eines KMS-Schlüssels aus.

iii. Wählen Sie unter Bucket Key (Bucket-Schlüssel) die Option Enable (Aktivieren). Die Copy-Operation wendet einen S3-Bucket-Schlüssel auf den Ziel-Bucket an.

8. Geben Sie Ihrem Job eine Beschreibung (oder behalten Sie die Standardeinstellung bei), legen Sie die Prioritätsstufe fest, wählen Sie einen Berichtstyp aus und geben Sie den Pfad zum Ziel des Fertigstellungsberichts an.
9. Stellen Sie im Abschnitt Permissions (Berechtigungen) sicher, dass Sie die IAM-Rolle für Batch-Vorgänge auswählen, die Sie zuvor definiert haben. Wählen Sie Next.
10. Überprüfen Sie die Einstellungen unter Review (Überprüfen). Wenn Sie Änderungen vornehmen müssen, wählen Sie Previous (Vorherige) aus. Nachdem Sie die Einstellungen für Batch-Vorgänge bestätigt haben, wählen Sie Create Job (Auftrag erstellen).

    Weitere Informationen finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags.

Ausführen eines Auftrags für S3-BatchVorgänge

Der Setup-Assistent kehrt Sie automatisch zum Abschnitt S3-Batch-Vorgänge der Amazon S3-Konsole zurück. Ihr neuer Auftrag wechselt vom New-Zustand in den Preparing-Zustand, da S3 den

Prozess beginnt. Während des Preparing-Zustands liest S3 das Manifest des Auftrags, prüft es auf Fehler und berechnet die Anzahl der Objekte.

1. Klicken Sie auf die Aktualisierungs-Schaltfläche in der Amazon S3-Konsole, um den Fortschritt zu überprüfen. Abhängig von der Größe des Manifests kann das Lesen Minuten oder Stunden dauern.
2. Nachdem S3 das Manifest des Auftrags gelesen hat, wechselt der Auftrag in den Zustand Awaiting your confirmation (Wartet auf Ihre Bestätigung). Klicken Sie auf die Options-Schaltfläche links neben der Auftrags-ID und wählen Sie Run job (Auftrag ausführen).
3. Wählen Sie die Einstellungen für den Auftrag aus und wählen Sie in der unteren rechten Ecke des Fensters Run job (Auftrag ausführen) aus.

   Nachdem der Auftrag ausgeführt wird, können Sie die Schaltfläche Aktualisieren auswählen, um den Fortschritt über die Dashboard-Ansicht der Konsole zu überprüfen oder den bestimmten Auftrag auszuwählen.
4. Wenn der Auftrag abgeschlossen wurde, können Sie die Successful (erfolgreichen) und Failed (fehlgeschlagenen) Objektzählungen anzeigen, um zu bestätigen, dass alles wie erwartet ausgeführt wurde. Wenn Sie Auftragsberichte aktiviert haben, überprüfen Sie Ihren Auftragsbericht auf die genaue Ursache für fehlgeschlagene Vorgänge.

   Sie können diese Schritte auch mithilfe der AWS CLI, AWS SDKs oder Amazon-S3-REST-API ausführen. Weitere Informationen zum Nachverfolgen von Auftragsstatus- und Abschlussberichten finden Sie unter Verfolgen von Auftragsstatus- und Abschluss.

Wissenswertes

Berücksichtigen Sie die folgenden Probleme, wenn Sie S3 Batch Operations verwenden, um Objekte mit S3-Bucket-Schlüsseln zu verschlüsseln:

- Die Kosten für Aufträge, Objekte und Anforderungen in S3 Batch Operations werden Ihnen zusätzlich zu allen mit der Operation, die S3 Batch Operations in Ihrem Namen ausführt, verbundenen Kosten berechnet, einschließlich Datenübertragungen, Anforderungen und anderen Gebühren. Weitere Informationen finden Sie unter Amazon S3 – Preise.
- Wenn Sie einen versionierten Bucket verwenden, erstellt jeder ausgeführte S3-Batchoperations-Auftrag neue verschlüsselte Versionen Ihrer Objekte. Darüber hinaus werden die vorherigen Versionen ohne konfigurierten S3-Bucket-Schlüssel beibehalten. Richten Sie zum Löschen der

alten Versionen eine S3-Lebenszyklus-Ablauf-Richtlinie für nicht aktuelle Versionen ein, wie unter Elemente der Lebenszyklus-Konfiguration beschrieben.
- Mit dem Kopiervorgang werden neue Objekte mit neuen Erstellungsdaten erstellt, die sich auf Lebenszyklus-Aktionen wie die Archivierung auswirken können. Wenn Sie alle Objekte in Ihrem Bucket kopieren, weisen alle neuen Kopien identische oder ähnliche Erstellungsdaten auf. Um diese Objekte weiter zu identifizieren und unterschiedliche Lebenszyklus-Regeln für verschiedene Daten-Teilmengen zu erstellen, sollten Sie Objekt-Markierungen verwenden.

Übersicht

In diesem Abschnitt sortierten Sie vorhandene Objekte, um bereits verschlüsselte Daten herauszufiltern. Anschließend haben Sie die Funktion S3-Bucket-Schlüssel auf unverschlüsselte Objekte angewendet, indem Sie mithilfe von S3 Batch Operations vorhandene Daten in einen Bucket mit aktiviertem S3-Bucket-Schlüssel kopiert haben. Dieser Prozess kann Ihnen Zeit und Geld sparen, während Sie Vorgänge wie die Verschlüsselung aller vorhandenen Objekte abschließen können.

Weitere Informationen über S3-Batch-Vorgänge finden Sie unter Ausführung umfangreicher Batch-Vorgänge für Amazon S3-Objekte durch..

Beispiele, die den Kopiervorgang mit Markierungen unter Verwendung der AWS CLI und AWS SDK for Java zeigen finden Sie unter Erstellen eines Batchoperations-Auftrags mit Auftrags-Markierungen zur Kennzeichnung.

Aufrufen einer AWS Lambda-Funktion

Die Invoke-AWS Lambda-Funktion initiiert AWS Lambda-Funktionen, um benutzerdefinierte Aktionen für Objekte auszuführen, die in einem Manifest aufgeführt sind. In diesem Abschnitt wird beschrieben, wie Sie eine Lambda-Funktion zur Verwendung mit S3-Batch-Vorgänge erstellen und wie Sie einen Auftrag zum Aufrufen der Funktion erstellen. Die S3-Batch-Vorgängeaufgabe verwendet die `LambdaInvoke`-Operation, um eine Lambda-Funktion für jedes in einem Manifest aufgeführte Objekt auszuführen.

Sie können mit S3-Batch-Vorgänge für Lambda über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder REST-APIs arbeiten. Weitere Informationen zur Verwendung von Lambda finden Sie unter Getting started with AWS Lambda (Erste Schritte mit Lambda) im AWS Lambda-Entwicklerhandbuch.

In den folgenden Abschnitten werden die ersten Schritte mit S3-Batch-Vorgänge mit Lambda erläutert.

Themen

Verwenden von Lambda mit Amazon S3-BatchVorgänge

Wenn Sie S3-Batch-Vorgänge mit AWS Lambda verwenden, müssen Sie neue Lambda-Funktionen speziell für die Verwendung mit S3-Batch-Vorgänge erstellen. Sie können vorhandene ereignisbasierte Funktionen von Amazon S3 nicht mit S3-Batch-Vorgänge wiederverwenden. Ereignisfunktionen können Nachrichten lediglich erhalten und nicht zurückgeben. Die Lambda-Funktionen, die mit S3-Batch-Vorgänge verwendet werden, müssen Nachrichten akzeptieren und zurückgeben. Weitere Informationen zur Verwendung von Lambda mit Amazon S3-Ereignissen finden Sie unter Using AWS Lambda with Amazon S3 (Verwendung von Lambda mit Amazon S3) im AWS Lambda-Entwicklerhandbuch.

Sie erstellen einen S3-Batchoperations-Auftrag, der Ihre Lambda-Funktion aufruft. Der Auftrag führt dieselbe Lambda-Funktion für alle in Ihrem Manifest aufgeführten Objekte aus. Sie können steuern, welche Versionen Ihrer Lambda-Funktion während der Verarbeitung der Objekte in Ihrem Manifest verwendet werden sollen. S3-Batch-Vorgänge unterstützt unqualifizierte Amazon-Ressourcennamen (ARNs), Aliasse und bestimmte Versionen. Weitere Informationen finden Sie unter Introduction to AWS Lambda Versioning (Einführung in die Lambda-Versionierung) im AWS Lambda-Entwicklerhandbuch.

Wenn Sie dem S3-Batchoperations-Auftrag mit einem Funktion-ARN zur Verfügung stellen, der einen Alias oder den-`$LATEST` Qualifizierer verwendet, und dann die Version aktualisieren, auf die einer davon verweist, beginnt S3-Batch-Vorgänge mit dem Aufruf der neuen Version Ihrer Lambda-Funktion. Dies kann nützlich sein, wenn Sie mitten in der Ausführung einer großen Aufgabe Funktionen aktualisieren möchten. Wenn S3-Batch-Vorgänge die verwendete Version nicht ändern soll, geben Sie die spezifische Version im Parameter `FunctionARN` an, wenn Sie den Auftrag erstellen.

Verwenden von Lambda und Amazon-S3-Stapelvorgängen mit Verzeichnis-Buckets.

Verzeichnis-Buckets sind eine Art von Amazon-S3-Buckets, die für Workloads oder leistungskritische Anwendungen entwickelt wurden, die eine konstante Latenzzeit im einstelligen Millisekundenbereich erfordern. Weitere Informationen finden Sie unter Verzeichnis-Buckets.

Für die Verwendung von Amazon-S3-Stapelvorgängen zum Aufrufen von Lambda-Funktionen, die auf Verzeichnis-Buckets wirken, gelten besondere Anforderungen. Beispielsweise müssen Sie Ihre Lambda-Anfrage mithilfe eines aktualisierten JSON-Schemas strukturieren und `InvocationSchemaVersion` **`2.0`** angeben, wenn Sie den Auftrag erstellen. Mit diesem aktualisierten Schema können Sie optionale Schlüssel-Wert-Paare für `UserArguments` angeben, mit denen Sie bestimmte Parameter vorhandener Lambda-Funktionen ändern können. Weitere Informationen finden Sie unter Automatisieren der Objektverarbeitung in Amazon S3-Verzeichnis-Buckets mit S3-Batchoperationen und AWS Lambda im AWS Storage Blog .

Antwort- und Ergebniscodes

Es gibt zwei Ebenen von Codes, die S3-Batch-Vorgänge von Lambda-Funktionen erwartet. Die erste ist der Antwortcode für die gesamte Anforderung, und die zweite ist ein Ergebniscode pro Aufgabe. Die folgende Tabelle enthält die Antwortcodes.

Antwortcode	Beschreibung
`Succeeded`	Die Aufgabe wurde normal abgeschlossen. Wenn Sie einen Aufgabenabschlussbericht angefordert haben, ist die Ergebniszeichenfolge der Aufgabe in dem Bericht enthalten.
`TemporaryFailure`	Die Aufgabe unterlag einem vorübergehenden Fehler und wird erneut ausgeführt, bevor der Job abgeschlossen ist. Die Ergebniszeichenfolge wird ignoriert. Wenn dies der letzte Versuch ist, ist die Fehlermeldung im abschließenden Bericht enthalten.
`PermanentFailure`	Die Aufgabe unterlag einem dauerhaften Fehler. Wenn Sie einen Aufgabenabschlussbericht angefordert haben, ist die Aufgabe als `Failed` markiert und enthält die Fehlermel

Antwortcode	Beschreibung
	dungszeichenfolge. Ergebniszeichenfolge aus fehlgeschlagenen Aufgaben werden ignoriert.

Erstellen einer Lambda-Funktion zur Verwendung mit S3-BatchVorgänge

Dieser Abschnitt enthält AWS Identity and Access Management (IAM) Beispiels-Berechtigungen, die Sie mit der Lambda-Funktion verwenden müssen. Er enthält auch eine Beispiel-Lambda-Funktion zur Verwendung mit S3-BatchVorgänge. Wenn Sie noch nie eine Lambda-Funktion erstellt haben, lesen Sie das Tutorial: Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda-Entwicklerhandbuch.

Sie müssen Lambda-Funktionen speziell für die Verwendung mit S3-Batch-Vorgänge erstellen. Sie können vorhandene ereignisbasierte Amazon S3-Lambda-Funktionen nicht wiederverwenden. Dies liegt daran, dass Lambda-Funktionen, die für S3-Batch-Vorgänge verwendet werden, spezielle Datenfelder akzeptieren und zurückgeben müssen.

> **Important**
>
> AWS Lambda In Java geschriebene Funktionen akzeptieren entweder - RequestHandler oder RequestStreamHandler -Handler-Schnittstellen. Um das Anforderungs- und Antwortformat von S3-Batch-Vorgänge zu unterstützen, benötigt AWS Lambda jedoch die `RequestStreamHandler`-Schnittstelle für die benutzerdefinierte Serialisierung und Deserialisierung von Anfrage und Antwort. Diese Schnittstelle ermöglicht es Lambda, ein InputStream und an die Java-`handleRequest`Methode OutputStream zu übergeben. Achten Sie darauf, die `RequestStreamHandler`-Schnittstelle zu verwenden, wenn Sie Lambda-Funktionen mit S3-Batch-Vorgänge verwenden. Wenn Sie eine `RequestHandler`-Schnittstelle verwenden, schlägt der Batchauftrag mit „Invalid JSON returned in Lambda payload (Ungültiges JSON in Lambda-Nutzlast zurückgegeben)“ im Abschlussbericht fehl. Weitere Informationen finden Sie unter Handler interfaces (Handler-Schnittstellen) im AWS Lambda-Benutzerhandbuch.

Beispiele für IAM-Berechtigungen

Im Folgenden finden Sie Beispiele für die IAM-Berechtigungen, die für die Verwendung einer Lambda-Funktion mit S3-Batch-Vorgänge erforderlich sind.

Example – Vertrauensrichtlinie für S3-BatchVorgänge

Nachfolgend finden Sie ein Beispiel für eine Vertrauensrichtlinie, die Sie für die Batchoperations-IAM-Rolle verwenden können. Diese IAM-Rolle wird beim Erstellen des Auftrags angegeben und erteilt Batch-Vorgänge die Berechtigung, die IAM-Rolle zu übernehmen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "batchoperations.s3.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Example – Lambda IAM-Richtlinie

Nachfolgend finden Sie ein Beispiel für eine IAM-Richtlinie, die S3-Batch-Vorgänge die Berechtigung zum Aufrufen der Lambda-Funktion und zum Lesen des Eingabemanifests erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "BatchOperationsLambdaPolicy",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:PutObject",
                "lambda:InvokeFunction"
            ],
            "Resource": "*"
        }
    ]
}
```

Beispiel: Anforderung und Antwort

Dieser Abschnitt bietet Beispiele für Anforderungen und Antworten für die Lambda-Funktion.

Example Anforderung

Nachfolgend sehen Sie ein JSON-Beispiel für eine Anforderung für die Lambda-Funktion.

```
{
    "invocationSchemaVersion": "1.0",
    "invocationId": "YXNkbGZqYWRmaiBhc2RmdW9hZHNmZGpmaGFzbGtkaGZza2RmaAo",
    "job": {
        "id": "f3cc4f60-61f6-4a2b-8a21-d07600c373ce"
    },
    "tasks": [
        {
            "taskId": "dGFza2lkZ29lc2hlcmUK",
            "s3Key": "customerImage1.jpg",
            "s3VersionId": "1",
            "s3BucketArn": "arn:aws:s3:us-east-1:0123456788:awsexamplebucket1"
        }
    ]
}
```

Example Antwort

Nachfolgend sehen Sie ein JSON-Beispiel für eine Antwort für die Lambda-Funktion.

```
{
  "invocationSchemaVersion": "1.0",
  "treatMissingKeysAs" : "PermanentFailure",
  "invocationId" : "YXNkbGZqYWRmaiBhc2RmdW9hZHNmZGpmaGFzbGtkaGZza2RmaAo",
  "results": [
    {
      "taskId": "dGFza2lkZ29lc2hlcmUK",
      "resultCode": "Succeeded",
      "resultString": "[\"Mary Major", \"John Stiles\"]"
    }
  ]
}
```

Beispiel für eine Lambda-Funktion für S3-BatchVorgänge

Im folgenden Beispiel entfernt Python Lambda eine Löschmarkierung von einem versionierten Objekt.

Wie das Beispiel zeigt, sind Schlüssel aus S3-Batch-Vorgänge URL-codiert. Um Amazon S3 mit anderen AWS-Services zu verwenden, ist es wichtig, dass Sie den Schlüssel, der von S3-Batch-Vorgänge übergeben wird, URL-dekodiert wird.

```
import logging
from urllib import parse
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
logger.setLevel("INFO")

s3 = boto3.client("s3")


def lambda_handler(event, context):
    """
    Removes a delete marker from the specified versioned object.

    :param event: The S3 batch event that contains the ID of the delete marker
                  to remove.
    :param context: Context about the event.
    :return: A result structure that Amazon S3 uses to interpret the result of the
             operation. When the result code is TemporaryFailure, S3 retries the
             operation.
    """
    # Parse job parameters from Amazon S3 batch operations
    invocation_id = event["invocationId"]
    invocation_schema_version = event["invocationSchemaVersion"]

    results = []
    result_code = None
    result_string = None

    task = event["tasks"][0]
    task_id = task["taskId"]

    try:
        obj_key = parse.unquote(task["s3Key"], encoding="utf-8")
        obj_version_id = task["s3VersionId"]
        bucket_name = task["s3BucketArn"].split(":")[-1]

        logger.info(
```

```
            "Got task: remove delete marker %s from object %s.", obj_version_id,
obj_key
        )

        try:
            # If this call does not raise an error, the object version is not a delete
            # marker and should not be deleted.
            response = s3.head_object(
                Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
            )
            result_code = "PermanentFailure"
            result_string = (
                f"Object {obj_key}, ID {obj_version_id} is not " f"a delete marker."
            )

            logger.debug(response)
            logger.warning(result_string)
        except ClientError as error:
            delete_marker = error.response["ResponseMetadata"]["HTTPHeaders"].get(
                "x-amz-delete-marker", "false"
            )
            if delete_marker == "true":
                logger.info(
                    "Object %s, version %s is a delete marker.", obj_key,
obj_version_id
                )
                try:
                    s3.delete_object(
                        Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
                    )
                    result_code = "Succeeded"
                    result_string = (
                        f"Successfully removed delete marker "
                        f"{obj_version_id} from object {obj_key}."
                    )
                    logger.info(result_string)
                except ClientError as error:
                    # Mark request timeout as a temporary failure so it will be
retried.
                    if error.response["Error"]["Code"] == "RequestTimeout":
                        result_code = "TemporaryFailure"
                        result_string = (
                            f"Attempt to remove delete marker from  "
                            f"object {obj_key} timed out."
```

```
                    )
                    logger.info(result_string)
                else:
                    raise
        else:
            raise ValueError(
                f"The x-amz-delete-marker header is either not "
                f"present or is not 'true'."
            )
    except Exception as error:
        # Mark all other exceptions as permanent failures.
        result_code = "PermanentFailure"
        result_string = str(error)
        logger.exception(error)
    finally:
        results.append(
            {
                "taskId": task_id,
                "resultCode": result_code,
                "resultString": result_string,
            }
        )
    return {
        "invocationSchemaVersion": invocation_schema_version,
        "treatMissingKeysAs": "PermanentFailure",
        "invocationId": invocation_id,
        "results": results,
    }
```

Erstellen eines S3-Batchoperations-Auftrags, der eine Lambda-Funktion aufruft

Wenn Sie einen S3-Batchoperations-Auftrag erstellen, um eine Lambda-Funktion aufzurufen, müssen Sie Folgendes angeben:

- Den ARN Ihrer Lambda-Funktion (möglicherweise mit dem Funktionsalias oder einer spezifischen Versionsnummer)
- Eine IAM-Rolle mit der Berechtigung zum Aufruf der Funktion
- Den Aktionsparamete `LambdaInvokeFunction`

Weitere Informationen zum Erstellen eines S3-Batchoperations-Auftrags finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags und Von S3 Batch-Vorgänge unterstützte Vorgänge.

Im folgenden Beispiel wird ein S3-Batchoperations-Auftrag erstellt, der eine Lambda-Funktion mit der AWS CLI CLI aufruft.

```
aws s3control create-job
    --account-id <AccountID>
    --operation  '{"LambdaInvoke": { "FunctionArn":
 "arn:aws:lambda:Region:AccountID:function:LambdaFunctionName" } }'
    --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key"]},"Location":
{"ObjectArn":"arn:aws:s3:::ManifestLocation","ETag":"ManifestETag"}}'
    --report
 '{"Bucket":"arn:aws:s3:::awsexamplebucket1","Format":"Report_CSV_20180820","Enabled":true,"Pre
    --priority 2
    --role-arn arn:aws:iam::AccountID:role/BatchOperationsRole
    --region Region
    --description “Lambda Function"
```

Bereitstellen von Informationen auf Aufgabenebene in Lambda-Manifesten

Wenn Sie AWS Lambda-Funktionen mit S3-Batch-Vorgänge verwenden, möchten Sie möglicherweise weitere Daten für die einzelnen Aufgaben/Schlüssel, die der Ausführung zugrunde liegen. Möglicherweise möchten Sie, dass sowohl ein Quellobjektschlüssel als auch ein neuer Objektschlüssel bereitgestellt wird. Ihre Lambda-Funktion kann in einem solchen Fall den Quellschlüssel in einen neuen S3-Bucket unter einem neuen Namen speichern. Standardmäßig können Sie mit Amazon S3-Batch-Vorgänge nur den Ziel-Bucket und eine Liste von Quellschlüsseln im Eingabemanifest für Ihren Auftrag angeben. Nachfolgend wird beschrieben, wie Sie weitere Daten in Ihr Manifest aufnehmen können, sodass Sie komplexere Lambda-Funktionen ausführen können.

Zum Angeben von Pro-Schlüssel-Parametern in Ihrem S3-BatchVorgängemanifest zur Verwendung im Code der Lambda-Funktion verwenden Sie das folgende URL-kodierte JSON-Format. Das `key`-Feld wird an Ihre Lambda-Funktion übergeben, als wäre es ein Amazon S3-Objektschlüssel. Es kann aber von der Lambda-Funktion so interpretiert werden, dass es andere Werte oder mehrere Schlüssel enthält, wie im Folgenden gezeigt:

> **Note**
>
> Die maximale Anzahl von Zeichen für das Feld `key` im Manifest beträgt 1.024.

Example – Manifest, dass die „Amazon S3-Schlüssel“ durch JSON-Zeichenfolgen ersetzt

Die URL-kodierte Version muss S3-Batch-Vorgänge bereitgestellt werden.

```
my-bucket,{"origKey": "object1key", "newKey": "newObject1Key"}
my-bucket,{"origKey": "object2key", "newKey": "newObject2Key"}
my-bucket,{"origKey": "object3key", "newKey": "newObject3Key"}
```

Example – URL-kodiertes Manifest

Diese URL-kodierte Version muss S3-Batch-Vorgänge bereitgestellt werden. Eine Version, die nicht URL-kodiert ist, wird nicht unterstützt.

```
my-bucket,%7B%22origKey%22%3A%20%22object1key%22%2C%20%22newKey%22%3A
%20%22newObject1Key%22%7D
my-bucket,%7B%22origKey%22%3A%20%22object2key%22%2C%20%22newKey%22%3A
%20%22newObject2Key%22%7D
my-bucket,%7B%22origKey%22%3A%20%22object3key%22%2C%20%22newKey%22%3A
%20%22newObject3Key%22%7D
```

Example – Lambda-Funktion im Manifest-Format, die Ergebnisse in den Auftragsbericht schreibt.

Diese Lambda-Funktion zeigt, wie eine durch Pipe getrennte Aufgabe analysiert wird, die im Amazon S3-BatchVorgängemanifest codiert ist. Die Aufgabe gibt an, welcher Revisionsvorgang auf das angegebene Objekt angewendet wird.

```
import logging
from urllib import parse
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
logger.setLevel("INFO")

s3 = boto3.resource("s3")


def lambda_handler(event, context):
    """
    Applies the specified revision to the specified object.

    :param event: The Amazon S3 batch event that contains the ID of the object to
                  revise and the revision type to apply.
```

```
    :param context: Context about the event.
    :return: A result structure that Amazon S3 uses to interpret the result of the
             operation.
    """
    # Parse job parameters from Amazon S3 batch operations
    invocation_id = event["invocationId"]
    invocation_schema_version = event["invocationSchemaVersion"]

    results = []
    result_code = None
    result_string = None

    task = event["tasks"][0]
    task_id = task["taskId"]
    # The revision type is packed with the object key as a pipe-delimited string.
    obj_key, revision = parse.unquote(task["s3Key"], encoding="utf-8").split("|")
    bucket_name = task["s3BucketArn"].split(":")[-1]

    logger.info("Got task: apply revision %s to %s.", revision, obj_key)

    try:
        stanza_obj = s3.Bucket(bucket_name).Object(obj_key)
        stanza = stanza_obj.get()["Body"].read().decode("utf-8")
        if revision == "lower":
            stanza = stanza.lower()
        elif revision == "upper":
            stanza = stanza.upper()
        elif revision == "reverse":
            stanza = stanza[::-1]
        elif revision == "delete":
            pass
        else:
            raise TypeError(f"Can't handle revision type '{revision}'.")

        if revision == "delete":
            stanza_obj.delete()
            result_string = f"Deleted stanza {stanza_obj.key}."
        else:
            stanza_obj.put(Body=bytes(stanza, "utf-8"))
            result_string = (
                f"Applied revision type '{revision}' to " f"stanza {stanza_obj.key}."
            )

        logger.info(result_string)
```

```
        result_code = "Succeeded"
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchKey":
            result_code = "Succeeded"
            result_string = (
                f"Stanza {obj_key} not found, assuming it was deleted "
                f"in an earlier revision."
            )
            logger.info(result_string)
        else:
            result_code = "PermanentFailure"
            result_string = (
                f"Got exception when applying revision type '{revision}' "
                f"to {obj_key}: {error}."
            )
            logger.exception(result_string)
    finally:
        results.append(
            {
                "taskId": task_id,
                "resultCode": result_code,
                "resultString": result_string,
            }
        )
    return {
        "invocationSchemaVersion": invocation_schema_version,
        "treatMissingKeysAs": "PermanentFailure",
        "invocationId": invocation_id,
        "results": results,
    }
```

Lernen mit einem Tutorial zu S3-Batchoperationen

Im folgenden Tutorial werden vollständige end-to-end Verfahren für einige Batchoperationenaufgaben mit Lambda vorgestellt.

- Tutorial: Batch-Transcodierung von Videos mit S3- AWS Lambda Batchoperationen und AWS Elemental MediaConvert

Alle Objektmarkierungen ersetzen

Der Vorgang Replace all object tags (Alle Objektmarkierungen ersetzen) ersetzt die Amazon S3-Objektmarkierungen für jedes im Manifest aufgeführte Objekt. Ein Amazon S3-Objekt-Tag ist ein Schlüssel-Wert-Paar, mit dem Sie Metadaten über ein Objekt speichern können.

Um eine Aufgabe „Alle Objektmarkierungen ersetzen“ zu erstellen, geben Sie eine Reihe von Markierungen an, die Sie anwenden möchten. S3 Batch-Vorgänge wendet auf jedes Objekt den gleichen Markierungssatz an. Der von Ihnen bereitgestellte Tag-Satz ersetzt alle Tag-Sätze, die bereits mit den Objekten im Manifest verknüpft sind. S3-Batch-Vorgänge unterstützen nicht das Hinzufügen von Markierungen zu Objekten, während die vorhandenen Markierungen unverändert gelassen werden.

Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, können Sie die Tag-Gruppen auf bestimmte Versionen der einzelnen Objekte anwenden. Hierzu geben Sie für jedes Objekt im Manifest eine Versions-ID an. Wenn Sie für ein Objekt keine Versions-ID angeben, wendet S3-Batch-Vorgänge den Tag auf die aktuelle Version jedes Objekts an.

Beschränkungen und Einschränkungen

- Die AWS Identity and Access Management (IAM)-Rolle, die Sie zum Ausführen des Batchoperations-Auftrags angeben, muss über Berechtigungen zum Durchführen der zugrunde liegenden Amazon S3-Operation „Alle Objektmarkierungen ersetzen“ verfügen. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter PutObjectTagging in der API-Referenz zum Amazon Simple Storage Service.
- S3-Batch-Vorgänge verwenden die Amazon S3 PutObjectTagging-Operation, um Markierungen auf jedes Objekt im Manifest anzuwenden. Alle Einschränkungen und Begrenzungen, die auf den zugrunde liegenden Vorgang zutreffen, gelten auch für S3-Batchoperations-Aufträge.

Weitere Informationen zur Verwendung der Konsole zum Erstellen von Aufträgen finden Sie unter Erstellen eines S3-Batchoperationsauftrags.

Weitere Informationen zur Objektmarkierung finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen in diesem Handbuch und unter PutObjectTagging, GetObjectTaggingund DeleteObjectTagging in der API-Referenz zum Amazon Simple Storage Service.

Alle Objektmrkierungen löschen

Der Vorgang Alle Objekt-Tags löschen entfernt alle Amazon S3-Objekt-Tag-Sätze, die derzeit mit den im Manifest aufgeführten Objekten verknüpft sind. S3 Batch-Vorgänge unterstützt nicht das Löschen von Markierungen aus Objekten, während andere Markierungen beibehalten werden.

Wenn sich die Objekte in Ihrem Manifest in einem versionierten Bucket befinden, können Sie die Tag-Sätze aus einer bestimmten Version eines Objekts entfernen. Hierzu geben Sie für jedes Objekt im Manifest eine Versions-ID an. Wenn Sie keine Versions-ID für ein Objekt angeben, entfernt S3 Batch-Vorgänge den Tag-Satz aus der neuesten Version jedes Objekts.

Weitere Informationen über Batchoperationsmanifeste finden Sie unter Angeben eines Manifests.

> **Warning**
>
> Durch die Ausführung dieser Aufgabe werden alle Objekt-Tag-Sätze für jedes im Manifest aufgelistete Objekt entfernt.

Beschränkungen und Einschränkungen

- Die AWS Identity and Access Management (IAM) Rolle, die Sie zum Ausführen der Aufgabe angeben, muss über Berechtigungen zum Durchführen des zugrunde liegenden Amazon S3-Vorgangs zur Löschung von Objektmarkierungen verfügen. Weitere Informationen finden Sie unter DeleteObjectTagging in der API-Referenz zum Amazon Simple Storage Service.
- S3 Batch-Vorgänge verwendet den Amazon S3-Vorgang DeleteObjectTagging, um die Markierungssätze von jedem Objekt im Manifest zu entfernen. Alle Einschränkungen und Begrenzungen, die auf den zugrunde liegenden Vorgang zutreffen, gelten auch für S3-Batchoperations-Aufträge.

Informationen zum Erstellen von Aufgaben finden Sie unter Erstellen eines S3-Batch-Vorgangsauftrags.

Weitere Informationen zur Objektmarkierung finden Sie unter Alle Objektmarkierungen ersetzen in diesem Handbuch und unter PutObjectTagging, GetObjectTaggingund DeleteObjectTagging in der API-Referenz zum Amazon Simple Storage Service.

Ersetzen der Access Control List

Der Vorgang Access Control List (ACL) ersetzen ersetzt die Amazon S3-Access-Control-Lists (ACLs) für jedes Objekt, das im Manifest aufgeführt ist. Mithilfe von ACLs können Sie festlegen, wer auf ein Objekt zugreifen darf, und welche Aktionen die Person ausführen darf.

S3-Batch-Vorgänge unterstützt benutzerdefinierte ACLs, die von ihnen festgelegt werden, sowie vorgefertigte ACLs, die von Amazon S3 mit vordefinierten Gruppen von Zugriffsberechtigungen bereitgestellt werden.

Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, können Sie die ACLs auf bestimmten Versionen der einzelnen Objekte anwenden. Hierzu geben Sie für jedes Objekt im Manifest eine Versions-ID an. Wenn Sie für ein Objekt keine Versions-ID angeben, wendet S3-Batch-Vorgänge die ACL auf die aktuelle Version des Objekts an.

Weitere Informationen zu ACLs in Amazon S3 erhalten Sie unt3, Zugriffskontrolllisten (ACL) – Übersicht.

S3 Block Public Access

Wenn Sie den öffentlichen Zugriff auf alle Objekte in einem Bucket einschränken möchten, sollten Sie die Amazon S3-Block-Einstellungen für den öffentlichen Zugriff anstelle von S3-Batch-Vorgänge verwenden. Block Public Access kann den öffentlichen Zugriff auf einen Bucket oder ein Konto mithilfe einer einzelnen, einfachen Operation einschränken, die sich zudem auch noch schnell umsetzen lässt. Daher ist dies die bessere Wahl, wenn Sie die Absicht haben, den öffentlichen Zugriff auf alle Objekte in einem Bucket oder einem Konto einzuschränken. Verwenden Sie S3-BatchVorgänge, wenn Sie eine benutzerdefinierte ACL auf jedes Objekt im Manifest anwenden möchten. Weitere Informationen über die S3 Block Public Access finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

S3 Object Ownership

Wenn sich die Objekte im Manifest in einem Bucket befinden, der die erzwungene Einstellung des Bucket-Eigentümers für „Object Ownership“ verwendet, kann die Operation Replace access control list (ACL) (Zugriffskontrollliste (ACL)) ersetzen nur Objekt-ACLs angeben, die dem Bucket-Eigentümer die volle Kontrolle gewähren. Der Vorgang kann anderen keine Objekt-ACL-Berechtigungen für AWS-Konten oder Gruppen erteilen. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Beschränkungen und Einschränkungen

- Die Rolle, die Sie zum Ausführen der Aufgabe „Access Control List ersetzen“ angeben, muss über Berechtigungen zum Durchführen des zugrunde liegenden Amazon S3-Vorgangs `PutObjectAcl` verfügen. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter PutObjectAcl in der API-Referenz zum Amazon Simple Storage Service.
- S3-Batch-Vorgänge verwenden die Amazon S3 `PutObjectAcl`-Operation, um die angegebene ACL auf jedes Objekt im Manifest anzuwenden. Daher gelten alle Einschränkungen und Begrenzungen, die auf den zugrunde liegenden PutObjectAcl-Vorgang zutreffen, auch für die S3-Batchoperations-Aufträge „Access Control List ersetzen“.

Objekte mit Batch Operations wiederherstellen

Der Wiederherstellungsvorgang initiiert Wiederherstellungsanforderungen für die archivierten Amazon-S3-Objekte, die in Ihrem Manifest aufgeführt sind. Die folgenden archivierten Objekte müssen wiederhergestellt werden, bevor auf sie in Echtzeit zugegriffen werden kann:

- Objekte, die in den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive archiviert werden
- Objekte, die über die Speicherklasse S3 Intelligent-Tiering in den Stufen Archive Access oder Deep Archive Access archiviert sind

Die Verwendung eines S3 Restore Object-Vorgangs in Ihrem S3 Batchoperations-Auftrag führt zu einer Wiederherstellungs-Anforderung für jedes im Manifest angegebene Objekt.

> **Important**
>
> Die S3-Aufgabe „Initiate Restore Object“ (Objektwiederherstellung initiieren) initiiert nur die Anfrage zum Wiederherstellen von Objekten. S3-Batch-Vorgänge melden die Aufgabe als abgeschlossen für jedes Objekt, nachdem die Anfrage für dieses Objekt initiiert wurde. Amazon S3 aktualisiert den Auftrag nicht und benachrichtigt Sie nicht weiter, wenn die Objekte wiederhergestellt wurden. Sie können jedoch S3-Ereignis-Benachrichtigungen verwenden, um Benachrichtigungen zu erhalten, wenn die Objekte in Amazon S3 verfügbar sind. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Wenn Sie die Aufgabe „S3 Initiate Restore Object“ erstellen, sind die folgenden Argumente verfügbar:

ExpirationInDays

Dieses Argument gibt an, wie lange das Objekt S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive in Amazon S3 verfügbar bleibt. Für „Initiate-Restore-Object-Aufgaben, die auf Objekte von S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive ausgelegt sind, muss `ExpirationInDays` auf 1 oder höher festgelegt werden.

> **Important**
>
> Legen Sie `ExpirationInDays` beim Erstellen von „S3 Initiate Restore Object“-Operationsaufgaben, die auf S3 Intelligent-Tiering Archive Access- und Deep Archive Access-Stufenobjekte ausgelegt sind, nicht fest. Objekte in S3 Intelligent-Tiering Archive Access-Stufen unterliegen keinem Ablaufdatum bei der Wiederherstellung, daher führt die Angabe von `ExpirationInDays` zu einem Fehler bei der Wiederherstellungsanforderung.

GlacierJobTier

Amazon S3 kann Objekte mithilfe einer von drei verschiedenen Abrufstufen wiederherstellen: `EXPEDITED`, `STANDARD` und `BULK`. Die S3-Batchoperations-Funktion unterstützt jedoch nur die Abrufstufen `STANDARD` und `BULK`. Weitere Informationen zu den Unterschieden zwischen den Wiederherstellungsstufen finden Sie unter Archiv-Abrufoptionen.

Weitere Informationen zu den Preisen für jede Stufe finden Sie im Abschnitt Anfragen und Datenabrufe auf der Seite Amazon S3-Preise.

Unterschiede bei der Wiederherstellung aus S3 Glacier und S3 Intelligent-Tiering

Das Wiederherstellen archivierter Dateien aus den Speicherklassen S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive unterscheidet sich von der Wiederherstellung von Dateien aus der Speicherklasse S3 Intelligent-Tiering in den Stufen Archive Access oder Deep Archive Access.

- Wenn Sie von S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive wiederherstellen, wird eine temporäre Kopie des Objekts erstellt. Amazon S3 löscht diese Kopie nachdem der Wert, den Sie im `ExpirationInDays`-Argument angegeben haben, abgelaufen ist. Nachdem diese temporäre Kopie gelöscht wurde, müssen Sie eine zusätzliche Wiederherstellungsanfrage einreichen, um auf das Objekt zugreifen zu können.

- Geben Sie bei der Wiederherstellung archivierter S3 Intelligent-Tiering-Objekte das `ExpirationInDays`-Argument nicht an. Wenn Sie ein Objekt aus den Stufen S3 Intelligent-Tiering Archive Access oder Deep Archive Access wiederherstellen, wird das Objekt zurück in die Speicherklasse S3 Intelligent-Tiering Frequent Access übergehen. Nach mindestens 90 aufeinanderfolgenden Tagen ohne Zugriff wechselt das Objekt automatisch in die Stufe „Archive Access“. Das Objekt wechselt nach mindestens 180 aufeinanderfolgenden Tagen ohne Zugriff automatisch in die Stufe „Deep Archive Access“.
- Batch-Operations-Aufträge können entweder mit Objekte der Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive oder mit Objekten der Speicherstufen S3 Intelligent-Tiering Archive Access und Deep Archive Access arbeiten. Batch Operations kann im selben Auftrag nicht für beide Typen von archivierten Objekten ausgeführt werden. Um Objekte beider Typen wiederherzustellen, müssen Sie separate Batchoperations-Aufgaben erstellen.

Überlappende Wiederherstellungen

Sollte die Aufgabe S3 Initiate Restore Object (Objektwiederherstellung initiieren) versuchen, ein Objekt wiederherzustellen, das gerade wiederhergestellt wird, geht S3 Batch-Vorgänge folgendermaßen vor:

Der Wiederherstellungsvorgang für das Objekt ist erfolgreich, wenn eine der folgenden Bedingungen erfüllt ist:

- Im Vergleich zu der bereits ausgeführten Wiederherstellungsanforderung ist der `ExpirationInDays`-Wert für diesen Auftrag identisch und sein `GlacierJobTier`-Wert ist schneller.
- Die vorherige Wiederherstellungsanforderung wurde bereits abgeschlossen und das Objekt ist derzeit verfügbar. In diesem Fall aktualisiert Batch-Vorgänge das Ablaufdatum des wiederhergestellten Objekts so, dass es mit dem in der laufenden Wiederherstellungsanforderung angegebenen `ExpirationInDays`-Wert übereinstimmt.

Der Wiederherstellungsvorgang für das Objekt schlägt fehl, wenn eine der folgenden Bedingungen erfüllt sind:

- Die bereits ausgeführte Wiederherstellungsanforderung wurde noch nicht abgeschlossen und die Wiederherstellungsdauer für diese Aufgabe (durch den `ExpirationInDays`-Wert angegeben) unterscheidet sich von der Wiederherstellungsdauer, die in der bereits ausgeführten Wiederherstellungsanforderung angegeben wurde.

- Die Wiederherstellungsebene für diesen Auftrag (durch den `GlacierJobTier`-Wert angegeben) ist identisch oder langsamer als die Wiederherstellungsebene, die in der bereits ausgeführten Wiederherstellungsanforderung angegeben wurde.

Einschränkungen

„S3 Initiate Restore Object“-Aufgaben haben folgende Einschränkungen:

- Sie müssen die Aufgabe in derselben Region wie die archivierten Objekte erstellen.
- S3 Batch-Vorgänge unterstützt die Abrufstufe `EXPEDITED` nicht.

Weitere Informationen zum Wiederherstellen von Objekten finden Sie unter Wiederherstellen eines archivierten Objekts.

Aufrechterhaltung der S3-Objektsperre

Der Vorgang Objektsperre zur Aufbewahrung ermöglicht es Ihnen, Aufbewahrungsdaten für Ihre Objekte entweder im Governance-Modus oder im Compliance-Modus anzuwenden. Diese Aufbewahrungsmodi wenden unterschiedliche Schutzgrade an. Sie können einen der beiden Aufbewahrungsmodi auf jede Objektversion anwenden. Aufbewahrungsdaten verhindern wie gesetzliche Aufbewahrungsfristen, dass ein Objekt überschrieben oder gelöscht wird. Amazon S3 speichert das in den Metadaten des Objekts angegebene Retain until (Aufbewahren bis)-Datum und schützt die angegebene Version der Objektversion bis zum Ablauf der Aufbewahrungsfrist.

Sie können S3-Batch-Vorgänge mit Objektsperre verwenden, um die Aufbewahrungsdaten vieler Amazon S3-Objekte gleichzeitig zu verwalten. Sie geben die Liste der Zielobjekte in Ihrem Manifest an und senden sie zur Fertigstellung an BatchVorgänge. Weitere Informationen finden Sie unter S3-Objektsperre the section called “Aufbewahrungszeiträume”.

Ihr S3-Batchoperations-Auftrag mit Aufbewahrungsdaten läuft bis zum Abschluss, bis zum Abbruch oder bis ein Fehlerstatus erreicht ist. Sie sollten S3-Batch-Vorgänge und die Aufbewahrung der S3-Objektsperre verwenden, wenn Sie das Aufbewahrungsdatum für viele Objekte mit einer einzigen Anfrage hinzufügen, ändern oder entfernen möchten.

Batch-Vorgänge überprüft, ob die Objektsperre in Ihrem Bucket aktiviert ist, bevor Schlüssel im Manifest verarbeitet werden. Um die Vorgänge und die Validierung durchzuführen, benötigt Batch-Vorgänge `s3:GetBucketObjectLockConfiguration`- und `s3:PutObjectRetention`-Berechtigungen in einer IAM-Rolle, damit Batch-Vorgänge die Objektsperre für Sie aufrufen kann. Weitere Informationen finden Sie unter the section called “Überlegungen zu Object Lock”.

Informationen zur Verwendung dieser Operation mit der REST-API finden Sie unter `S3PutObjectRetention` in der CreateJob-Operation in der Amazon Simple Storage Service API-Referenz.

Ein AWS Command Line Interface-Beispiel für die Verwendung dieser Operation finden Sie unter the section called “Batch-Vorgänge mit Objektsperrenaufbewahrung verwenden”. Ein AWS SDK for Java-Beispiel finden Sie unter the section called “Batch-Vorgänge mit Objektsperrenaufbewahrung verwenden”.

Beschränkungen und Einschränkungen

- S3-Batch-Vorgänge nimmt keine Änderungen auf Bucket-Ebene vor.
- Versionierung und die S3-Objektsperre müssen auf dem Bucket konfiguriert werden, in dem der Auftrag ausgeführt wird.
- Alle im Manifest aufgelisteten Objekte müssen sich im selben Bucket befinden.
- Die Operation funktioniert mit der neuesten Version des Objekts, es sei denn, eine Version ist explizit im Manifest angegeben.
- Sie benötigen eine `s3:PutObjectRetention`-Berechtigung in Ihrer IAM-Rolle, um dies zu verwenden.
- `s3:GetBucketObjectLockConfiguration`Die -IAM-Berechtigung ist erforderlich, um zu bestätigen, dass die Objektsperre für den S3-Bucket aktiviert ist.
- Sie können den Aufbewahrungszeitraum von Objekten nur verlängern, bei denen die Aufbewahrungsdaten des `COMPLIANCE`-Modus angewendet wurden, und er kann nicht verkürzt werden.

S3-Objektsperre aufgrund gesetzlicher Aufbewahrungsfristen

Die Objektsperre aufgrund gesetzlicher Aufbewahrungsfristen ermöglicht es Ihnen, eine Objektversion legal zu halten. Wie das Festlegen von Aufbewahrungszeiträumen verhindern auch rechtliche Aufbewahrungsfristen das Überschreiben oder Löschen von Objektversionen. Mit rechtlichen Aufbewahrungsfristen sind jedoch keine Aufbewahrungszeiträume verknüpft. Sie sind gültig, bis sie entfernt werden.

Sie können S3-Batch-Vorgänge mit Objektsperre verwenden, um vielen Amazon S3-Objekten gleichzeitig gesetzliche Aufbewahrungsfristen hinzuzufügen. Sie können dies tun, indem Sie die Zielobjekte in Ihrem Manifest auflisten und diese Liste an Batch-Vorgänge senden. Ihr S3-

Batchoperations-Auftrag mit Objektsperre für die gesetzliche Aufbewahrungsfrist bis zum Abschluss, bis zur Kündigung oder bis ein Ausfallstatus erreicht ist, ausgeführt.

S3-Batch-Vorgänge überprüft, ob die Objektsperre in Ihrem S3-Bucket aktiviert ist, bevor Schlüssel im Manifest verarbeitet werden. Um die Objekt-Vorgänge und die Validierung auf Bucket-Ebene durchzuführen, benötigt S3-Batch-Vorgänge `s3:PutObjectLegalHold` und `s3:GetBucketObjectLockConfiguration` in einer IAM-Rolle, die es S3-Batch-Vorgänge ermöglicht, die S3-Objektsperre für Sie aufzurufen.

Wenn Sie den S3-Batchoperations-Auftrag erstellen, um die gesetzliche Aufbewahrungsfrist zu entfernen, müssen Sie lediglich Off (Aus) als Status der gesetzlichen Aufbewahrungsfrist angeben. Weitere Informationen finden Sie unter the section called “Überlegungen zu Object Lock”.

Informationen zur Verwendung dieser Operation mit der REST-API finden Sie unter `S3PutObjectLegalHold` in der CreateJob-Operation in der API-Referenz zum Amazon Simple Storage Service.

Eine Beispielverwendung dieser Operation finden Sie unter Verwenden des AWS-SDK für Java.

Beschränkungen und Einschränkungen

- S3-Batch-Vorgänge nimmt keine Änderungen auf Bucket-Ebene vor.
- Alle im Manifest aufgelisteten Objekte müssen sich im selben Bucket befinden.
- Versionierung und die S3-Objektsperre müssen auf dem Bucket konfiguriert werden, in dem der Auftrag ausgeführt wird.
- Die Operation funktioniert mit der neuesten Version des Objekts, es sei denn, eine Version ist explizit im Manifest angegeben.
- `s3:PutObjectLegalHold`Die -Berechtigung ist in Ihrer IAM-Rolle erforderlich, um Objekten gesetzliche Aufbewahrungsfristen hinzuzufügen oder von diesen zu entfernen.
- `s3:GetBucketObjectLockConfiguration` Die IAM-Berechtigung ist erforderlich, um zu bestätigen, dass die S3-Objektsperre für den S3-Bucket aktiviert ist.

- Kopieren von Objekten
- Aufrufen einer AWS Lambda-Funktion
- Alle Objektmarkierungen ersetzen
- Alle Objektmrkierungen löschen

- Ersetzen der Access Control List
- Objekte mit Batch Operations wiederherstellen
- Aufrechterhaltung der S3-Objektsperre
- S3-Objektsperre aufgrund gesetzlicher Aufbewahrungsfristen
- Replizieren bestehender Objekte mit S3-Batch-Replikation

Verwalten von S3-Batch-Vorgangsaufträgen

Amazon S3 bietet robuste Tools, die Sie bei der Verwaltung Ihrer Aufträge für die S3-Batch-Vorgänge unterstützen, nachdem Sie diese erstellt haben. In diesem Abschnitt werden die Vorgänge beschrieben, mit denen Sie Ihre Aufträge mithilfe der AWS Management Console, AWS CLI, AWS-SDKs oder REST-API verwalten und verfolgen können.

Themen

- Verwenden der Amazon-S3-Konsole zum Verwalten Ihrer S3-Batch-Vorgangsaufträge
- Auflisten von Aufträgen
- Anzeigen von Auftragsdetails
- Zuweisen der Auftragspriorität

Verwenden der Amazon-S3-Konsole zum Verwalten Ihrer S3-Batch-Vorgangsaufträge

Verwenden der Konsole zum Verwalten Ihrer S3-Batch-Vorgangsaufträge Beispielsweise ist Folgendes möglich:

- Aktive und in Warteschlange gewartete Aufträge anzeigen
- Ändern der Priorität eines Auftrags
- Bestätigen und Ausführen eines Auftrags
- Einen Auftrag klonen
- Abbrechen eines Auftrags

Verwenden der Konsole zum Verwalten Ihrer Batchvorgänge

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Klicken Sie im linken Navigationsbereich auf Batchvorgänge.
3. Wählen Sie den spezifischen Auftrag aus, den Sie verwalten möchten.

Auflisten von Aufträgen

Sie können eine Liste Ihrer S3-Batch-Vorgangsaufträge abrufen. Die Liste umfasst Aufträge, die noch nicht abgeschlossen sind, sowie Aufträge, die in den letzten 90 Tagen abgeschlossen wurden. Die Auftragsliste enthält Informationen zu jedem Auftrag, z. B. die ID, die Beschreibung, die Priorität, den aktuellen Status und die Anzahl an Aufgaben, die erfolgreich durchgeführt wurden oder fehlgeschlagen sind. Sie können die Auftragsliste nach dem Status filtern. Wenn Sie eine Auftragsliste über die Konsole abrufen, können Sie Ihre Aufträge auch nach Beschreibung oder ID durchsuchen und nach AWS-Region filtern.

Abrufen einer Liste von aktiven und abgeschlossenen Aufträgen

Im folgenden AWS CLI-Beispiel wird eine Liste von `Active` und `Complete` Aufträgen abgerufen.

```
aws s3control list-jobs \
    --region us-west-2 \
    --account-id acct-id \
    --job-statuses '["Active","Complete"]' \
    --max-results 20
```

Weitere Informationen und Beispiele finden Sie unter list-jobs in der AWS CLI-Befehlsreferenz.

Anzeigen von Auftragsdetails

Wenn Sie mehr Informationen zu einem Auftrag benötigen, als Sie beim Auflisten von Aufträgen erhalten, können Sie alle Details zu einem einzelnen Auftrag anzeigen lassen. Sie können Details zu Aufträgen anzeigen, die noch nicht abgeschlossen sind, oder zu Aufträgen, die in den letzten 90 Tagen abgeschlossen wurden. Neben den Angaben, die in einer Auftragsliste enthalten sind, umfassen die Details zu einem einzelnen Auftrag auch noch andere Informationen, wie z. B:

- Die Betriebsparameter
- Details zum Manifest
- Informationen zum Abschlussbericht (falls Sie einen konfiguriert haben, als Sie den Auftrag erstellt haben)

- Der Amazon-Ressourcennamen (ARN) der Benutzerrolle, die Sie zum Ausführen des Auftrags zugewiesen haben

Durch das Anzeigen der Details eines einzelnen Auftrags erhalten Sie Zugriff auf die gesamte Konfiguration eines Auftrags. Sie können die Details zu einem Auftrag über die Amazon-S3-Konsole oder die AWS Command Line Interface (AWS CLI) anzeigen.

Die Beschreibung eines Auftrags in S3 Batch Operations in der Amazon-S3-Konsole abrufen

So zeigen Sie die Beschreibung eines Batch-Operations-Auftrags über die Konsole an

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie im linken Navigationsbereich auf Batchvorgänge.
3. Wählen Sie die Auftrags-ID des spezifischen Auftrags aus, um die zugehörigen Details anzuzeigen.

Die Beschreibung eines Auftrags in S3 Batch Operations in der AWS CLI abrufen

Im folgenden Beispiel wird die Beschreibung eines Auftrags in S3 Batch Operations unter Verwendung der AWS CLI abgerufen. Wenn Sie den folgenden Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

```
aws s3control describe-job \
--region us-west-2 \
--account-id acct-id \
--job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c
```

Weitere Informationen und Beispiele finden Sie unter describe-job in der AWS CLI-Befehlsreferenz.

Zuweisen der Auftragspriorität

Sie können jedem Auftrag eine numerische Priorität zuweisen, bei der es sich um eine beliebige positive Ganzzahl handeln kann. S3-Batch-Vorgänge priorisiert Aufträge entsprechend der zugewiesenen Priorität. Aufträge mit höherer Priorität (oder einem höheren Wert für den Prioritätsparameter) werden zuerst bewertet. Die Priorität wird in absteigender Reihenfolge ermittelt.

So erhält beispielsweise eine Auftragswarteschlange mit einem Prioritätswert von 10 Vorrang vor einer Auftragswarteschlange mit dem Wert 1.

Sie können die Priorität eines Auftrags auch während seiner Ausführung ändern. Wenn Sie einen neuen Auftrag übermitteln, während ein Auftrag ausgeführt wird, kann der Auftrag mit niedrigerer Priorität pausieren und der Auftrag mit höherer Priorität ausgeführt werden.

Das Ändern der Auftragspriorität hat keinen Einfluss auf die Geschwindigkeit der Auftragsverarbeitung.

> **Note**
>
> S3-Batch-Vorgänge berücksichtigt Auftragsprioritäten auf Best Effort-Basis. Zwar werden Aufträge mit höheren Prioritäten im Allgemeinen vor Aufträgen mit niedrigeren Prioritäten ausgeführt, Amazon S3 garantiert jedoch keine strikte Reihenfolge der Aufträge.

Verwenden der S3-Konsole

So aktualisieren Sie die Auftrags-Priorität in der AWS Management Console

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie im linken Navigationsbereich auf Batchvorgänge.
3. Wählen Sie den spezifischen Auftrag aus, den Sie verwalten möchten.
4. Wählen Sie Action (Aktion). Wählen Sie in der Dropdown-Liste Update priority (Priorität aktualisieren).

Verwendung der AWS CLI

Im folgenden Beispiel wird die Auftragspriorität mithilfe der aktualisier AWS CLI. Eine höhere Zahl weist auf eine höhere Ausführungspriorität hin.

```
aws s3control update-job-priority \
    --region us-west-2 \
    --account-id acct-id \
    --priority 98 \
    --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c
```

Verwendung der AWS SDK for Java

Im folgenden Beispiel wird die Priorität eines S3-Batch-Vorgangsauftrags unter Verwendung des AWS SDK for Java aktualisiert.

Weitere Informationen zur Auftragspriorität finden Sie unter Zuweisen der Auftragspriorität.

Example

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.UpdateJobPriorityRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateJobPriority {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String jobId = "00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.updateJobPriority(new UpdateJobPriorityRequest()
                    .withAccountId(accountId)
                    .withJobId(jobId)
                    .withPriority(98));

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
```

```
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verfolgen von Auftragsstatus- und Abschluss

Mit S3 Batch Operations können Sie den Auftragsstatus anzeigen und aktualisieren, Benachrichtigungen und Protokollierung hinzufügen, Auftragsfehler verfolgen und Abschlussberichte erstellen.

Themen

- Auftragsstatuses
- Auftragsstatus wird aktualisiert
- Benachrichtigungen und Protokollierung
- Nachverfolgen von Auftragsfehlern
- Abschlussberichte
- Beispiele: Nachverfolgen eines S3-Batchoperations-Auftrags in Amazon EventBridge über AWS CloudTrail
- Beispiele: Abschlussberichte zu S3-BatchVorgänge

Auftragsstatuses

Nachdem Sie einen Auftrag erstellt und ausgeführt haben, durchläuft dieser eine Reihe von Status. In der folgenden Tabelle werden der Status und die möglichen Übergänge zwischen ihnen beschrieben.

Status	Beschreibung	Übergänge
`New`	Ein Auftrag beginnt nach der Erstellung mit dem Status `New`.	Ein Auftrag wechselt automatisch in den Status `Preparing` , sobald Amazon S3 mit der Verarbeitung des Manifestobjekts beginnt.
`Preparing`	Amazon S3 verarbeitet das Manifestobjekt und andere	Ein Auftrag wechselt automatisch in den Status

Status	Beschreibung	Übergänge
	Auftragsparameter, um den Auftrag einzurichten und auszuführen.	`Ready`, sobald Amazon S3 die Verarbeitung des Manifests und anderer Parameter abschließt. Er ist dann bereit, die angegebene Operation an den im Manifest aufgeführten Objekten auszuführen. Wenn der Auftrag vor der Ausführung bestätigt werden muss, wie dies bei der Erstellung eines Auftrags mit der Amazon-S3-Konsole der Fall ist, wechselt der Auftrag von `Preparing` zu `Suspended` . Der Auftrag verbleibt im Status `Suspended` , bis Sie bestätigen, dass Sie ihn ausführen möchten.

Status	Beschreibung	Übergänge
`Suspended`	Der Auftrag erfordert eine Bestätigung, aber Sie haben noch nicht bestätigt, dass Sie ihn ausführen möchten. Nur Aufträge, die mit der Amazon-S3-Konsole erstellt wurden, benötigen eine Bestätigung. Ein Auftrag, der mit der Konsole erstellt wurde, wechselt nach `Suspended` sofort zu `Preparing` . Nachdem Sie die Ausführung des Auftrags bestätigen und der Auftrag in den Status `Ready` wechselt, kehrt er nicht wieder in den Status `Suspended` zurück.	Nachdem Sie die Ausführung des Auftrags bestätigen, wechselt der Status zu `Ready`.
`Ready`	Amazon S3 ist bereit, die angeforderten Objekt-Vorgänge auszuführen.	Ein Auftrag wechselt automatisch in den Status `Active`, sobald Amazon S3 mit dessen Ausführung beginnt. Der Zeitraum, den ein Auftrag im Status `Ready` verbringt, hängt davon ab, ob bereits Aufträge mit einer höheren Priorität ausgeführt werden und wie lange es dauert, diese Aufträge abzuschließen.

Status	Beschreibung	Übergänge
`Active`	Amazon S3 führt die angeforderte Operation an den im Manifest aufgeführten Objekten aus. Während ein Auftrag den Status `Active` hat, können Sie seinen Fortschritt mithilfe der Amazon-S3-Konsole oder der `DescribeJob` -Operation über die REST-API, AWS CLI, oder AWS-SDKs überwachen.	Ein Auftrag verlässt den Status `Active`, wenn er nicht länger Vorgänge an Objekten ausführt. Dies kann automatisch erfolgen, wenn ein Auftrag erfolgreich beendet wird oder fehlschlägt. Oder es kann aufgrund von Benutzeraktionen erfolgen, wenn beispielsweise ein Auftrag abgebrochen wird. Der neue Status des Auftrags hängt vom Grund für den Übergang ab.
`Pausing`	Der Auftrag wechselt von `Paused` zu einem anderen Status über.	Ein Auftrag wechselt automatisch in den Status `Paused`, wenn die Stufe `Pausing` abgeschlossen ist.
`Paused`	Ein Auftrag wechselt in den Status `Paused`, wenn ein anderer Auftrag mit einer höheren Priorität abgesendet wird, während der aktuelle Auftrag ausgeführt wird.	Ein Auftrag mit dem Status `Paused` kehrt automatisch in den Status `Active` zurück, nachdem alle Aufträge mit einer höheren Priorität, die die Ausführung des Auftrag blockierten, abgeschlossen, fehlgeschlagen oder gesperrt sind.

Status	Beschreibung	Übergänge
`Complete`	Der Auftrag hat die Ausführung der angeforderten Operation an allen Objekten abgeschlossen. Die Operation kann bei den einzelnen Objekten entweder erfolgreich abgeschlossen oder fehlgeschlagen sein. Wenn Sie den Auftrag so konfiguriert haben, dass ein Abschlussbericht generiert werden soll, dann ist der Bericht sofort verfügbar, nachdem der Auftrag in den Status gewechselt is `Complete`.	`Complete` ist ein Beendigungsstatus. Sobald ein Auftrag den Status `Complete` erreicht, geht er in keinen anderen Status über.
`Cancelling`	Der Auftrag wechselt in den Status `Cancelled` .	Ein Auftrag wechselt automatisch in den Status `Cancelled` , wenn die Stufe `Cancelling` abgeschlossen ist.
`Cancelled`	Sie haben angefordert, dass der Auftrag abgebrochen wird, und S3-Batch-Vorgänge hat den Auftrag erfolgreich abgebrochen. Der Auftrag sendet keine neuen Anforderungen an Amazon S3.	`Cancelled` ist ein Beendigungsstatus. Nachdem ein Auftrag den Status `Cancelled` erreicht, geht er in keinen anderen Status über.
`Failing`	Der Auftrag wechselt in den Status `Failed`.	Ein Auftrag wechselt automatisch in den Status `Failed`, nachdem die Stufe `Failing` abgeschlossen ist.

Status	Beschreibung	Übergänge
`Failed`	Der Auftrag ist fehlgesch lagen und wird nicht länger ausgeführt. Weitere Informati onen zu Auftragsfehlern finden Sie unter Nachverfolgen von Auftragsfehlern.	`Failed` ist ein Beendigun gsstatus. Nachdem ein Auftrag den Status `Failed` erreicht, geht er in keinen anderen Status über.

Auftragsstatus wird aktualisiert

Die folgenden Beispiele für AWS CLI und SDK für Java aktualisieren den Status eines Batch-Vorgangsauftrags. Weitere Informationen zur Verwendung der S3-Konsole zur Verwaltung von Batch-Vorgänge-Aufträgen finden Sie unter Verwenden der Amazon-S3-Konsole zum Verwalten Ihrer S3-Batch-Vorgangsaufträge.

Verwendung der AWS CLI

- Wenn Sie den Parameter `--no-confirmation-required` im vorherigen `create-job`-Beispiel nicht angegeben haben, bleibt der Auftrag so lange ausgesetzt, bis Sie den Auftrag bestätigen, indem Sie dessen Status auf `Ready` setzen. Amazon S3 erlaubt dann die Ausführung des Auftrags.

```
aws s3control update-job-status \
    --region us-west-2 \
    --account-id 181572960644 \
    --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c \
    --requested-job-status 'Ready'
```

- Brechen Sie den Auftrag ab, indem Sie den Auftragsstatus auf einstelle `Cancelled`.

```
aws s3control update-job-status \
     --region us-west-2 \
     --account-id 181572960644 \
     --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c \
     --status-update-reason "No longer needed" \
     --requested-job-status Cancelled
```

Verwenden des AWS-SDK für Java

Im folgenden Beispiel wird der Status eines S3-Batch-Vorgangsauftrags unter Verwendung des aktualisiert AWS SDK for Java.

Weitere Informationen zum Auftragsstatus finden Sie unter Verfolgen von Auftragsstatus- und Abschluss.

Example

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.UpdateJobStatusRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateJobStatus {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String jobId = "00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.updateJobStatus(new UpdateJobStatusRequest()
                    .withAccountId(accountId)
                    .withJobId(jobId)
                    .withRequestedJobStatus("Ready"));

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
```

```
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Benachrichtigungen und Protokollierung

Zusätzlich zum Anfordern von Abschlussberichten können Sie die Batch-Vorgängeaktivitäten auch mithilfe von erfassen, überprüfen und prüfe AWS CloudTrail. Da Batch-Vorgänge vorhandene Amazon-S3-APIs für die Durchführung von Aufgaben nutzt, geben diese Aufgaben dieselben Ereignisse aus, die sie ausgeben würden, wenn Sie sie direkt aufgerufen hätten. So können Sie den Fortschritt Ihres Auftrags und aller seiner Aufgaben mit denselben Benachrichtigungs-, Protokollierungs- und Prüftools und -prozessen nachverfolgen und aufzeichnen, die Sie bereits mit Amazon S3 verwenden. Weitere Informationen finden Sie in den Beispielen in den folgenden Abschnitten.

> **Note**
>
> Amazon S3 Batch-Vorgänge generiert während der Aufgabenausführung sowohl Management- als auch Datenereignisse in CloudTrail. Das Volumen dieser Ereignisse wird mit der Anzahl der Schlüssel im Manifest jeder Aufgabe skaliert. Weitere Informationen finden Sie auf der CloudTrail-Preisseite, die Beispiele dafür enthält, wie sich die Preise je nach Anzahl der in Ihrem Konto konfigurierten Trails ändern. Informationen zum Konfigurieren und Protokollieren von Ereignissen, damit sie Ihren Anforderungen entsprechen, finden Sie unter Create your first trail (Erstellen Ihres ersten Trails) im AWS CloudTrail-Benutzerhandbuch.

Weitere Informationen über Amazon-S3-Ereignisse finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Nachverfolgen von Auftragsfehlern

Wenn bei einem S3-Batch-Vorgangsauftrag ein Problem auftritt, das die erfolgreiche Ausführung verhindert, schlägt der Auftrag fehl. Dies ist beispielsweise der Fall, wenn das angegebene Manifest nicht gelesen werden kann. Wenn ein Auftrag fehlschlägt, generiert er einen oder mehrere Fehlercodes oder Fehlergründe. S3-Batch-Vorgänge speichert die Fehlercodes und die Gründe mit

dem Auftrag, damit Sie sie anzeigen können, indem Sie die Details des Auftrags anfordern. Wenn Sie einen Abschlussbericht für den Auftrag angefordert haben, sind die Fehlercodes und -ursachen auch darin enthalten.

Um zu verhindern, dass zu viele Vorgänge fehlschlagen, wendet Amazon S3 einen Schwellenwert für fehlgeschlagene Aufträge auf jeden Batch-Vorgangsauftrag an. Wenn ein Auftrag mindestens 1.000 Aufgaben ausgeführt hat, überwacht Amazon S3 die Aufgabenfehlerrate. Wenn die Fehlerrate (also die Anzahl fehlgeschlagener Aufgaben im Verhältnis zur Gesamtzahl ausgeführter Aufgaben) zu einem beliebigen Zeitpunkt 50 Prozent überschreitet, schlägt der Auftrag fehl. Wenn Ihr Auftrag fehlschlägt, weil er den Schwellenwert für fehlgeschlagene Aufträge überschritten hat, können Sie die Ursache der Fehler identifizieren. Beispielsweise könnte der Fehler darin liegen, dass Sie versehentlich Objekte in das Manifest aufgenommen haben, die im angegebenen Bucket nicht vorhanden sind. Nach dem Beheben der Fehler können Sie den Auftrag erneut übermitteln.

> **Note**
>
> S3-Batch-Vorgänge wird asynchron betrieben, und die Aufträge werden nicht zwangsläufig in der Reihenfolge ausführt, in der die Objekte im Manifest aufgeführt sind. Daher können Sie anhand der Reihenfolge im Manifest nicht feststellen, welche Objektaufgaben erfolgreich ausgeführt wurden und welche fehlgeschlagen sind. Um dies zu ermitteln, können Sie (sofern Sie ihn angefordert haben) den Abschlussbericht zu Ihrem Auftrag nutzen oder ihre AWS CloudTrail-Ereignisprotokolle anzeigen lassen, um so die Fehlerursache zu finden.

Abschlussberichte

Wenn Sie einen Auftrag erstellen, können Sie einen Abschlussbericht anfordern. Solange S3-Batch-Vorgänge mindestens eine Aufgabe erfolgreich aufruft, generiert Amazon S3 einen Abschlussbericht, wenn die Ausführung von Aufträgen abgeschlossen wurde, fehlgeschlagen ist oder abgebrochen wurde. Sie können den Abschlussbericht so konfigurieren, dass er alle oder nur fehlgeschlagene Aufgaben enthält.

Der Abschlussbericht umfasst neben der Auftragskonfiguration und dem Auftragsstatus auch Informationen zu jeder Aufgabe, darunter zu Objektschlüssel und -version, Status, Fehlercodes und Beschreibungen von Fehlern. Abschlussberichte bieten eine einfache Möglichkeit, die Ergebnisse von Aufgaben in einem konsolidierten Format ohne zusätzliche Einrichtung anzuzeigen. Ein Beispiel für einen Abschlussbericht finden Sie unter Beispiele: Abschlussberichte zu S3-BatchVorgänge.

Wenn Sie keinen Abschlussbericht konfigurieren, können Sie Ihren Auftrag und dessen Aufgaben mit CloudTrail und Amazon CloudWatch dennoch überwachen und prüfen. Weitere Informationen finden Sie im folgenden Abschnitt.

Themen

- Beispiele: Nachverfolgen eines S3-Batchoperations-Auftrags in Amazon EventBridge über AWS CloudTrail
- Beispiele: Abschlussberichte zu S3-BatchVorgänge

Beispiele: Nachverfolgen eines S3-Batchoperations-Auftrags in Amazon EventBridge über AWS CloudTrail

Die Auftragsaktivität von Amazon S3-Batch-Vorgänge wird in Form von Ereignissen in aufgezeichne AWS CloudTrail. Sie können eine benutzerdefinierte Regel in Amazon EventBridge erstellen und diese Ereignisse an die gewünschte Zielbenachrichtigungsressource senden, z. B. Amazon Simple Notification Service (Amazon SNS).

> **Note**
>
> Amazon EventBridge ist die bevorzugte Methode zum Verwalten Ihrer Ereignisse. Amazon CloudWatch Events und EventBridge liegen der gleiche Service und die gleiche API zugrunde, EventBridge bietet jedoch mehr Funktionen. Änderungen, die Sie in CloudWatch oder EventBridge vornehmen, werden in allen Konsolen angezeigt. Weitere Informationen finden Sie im Benutzerhandbuch für Amazon EventBridge.

Tracking-Beispiele

- In CloudTrail aufgezeichnete S3 Batchoperationsereignisse
- EventBridge-Regel zur Nachverfolgung von S3-Batchoperations-Auftragsereignissen

In CloudTrail aufgezeichnete S3 Batchoperationsereignisse

Wenn ein Batchoperations-Auftrag erstellt wird, wird er als `JobCreated`-Ereignis in CloudTrail aufgezeichnet. Während der Ausführung des Auftrags ändert sich der Status während der Verarbeitung und andere `JobStatusChanged`-Ereignisse werden in CloudTrail aufgezeichnet. Sie

können diese Ereignisse in der CloudTrail-Konsole anzeigen. Weitere Informationen zu CloudTrail finden Sie im AWS CloudTrail-Benutzerhandbuch.

> **Note**
>
> In CloudTrail werden nur `status-change`-Ereignisse zu S3-Batch-Vorgänge-Aufträgen aufgezeichnet.

Example In CloudTrail aufgezeichnetes S3-Batchoperations-Auftragsabschlussereignis

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "accountId": "123456789012",
        "invokedBy": "s3.amazonaws.com"
    },
    "eventTime": "2020-02-05T18:25:30Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "JobStatusChanged",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "s3.amazonaws.com",
    "userAgent": "s3.amazonaws.com",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "f907577b-bf3d-4c53-b9ed-8a83a118a554",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "123412341234",
    "serviceEventDetails": {
        "jobId": "d6e58ec4-897a-4b6d-975f-10d7f0fb63ce",
        "jobArn": "arn:aws:s3:us-west-2:181572960644:job/
d6e58ec4-897a-4b6d-975f-10d7f0fb63ce",
        "status": "Complete",
        "jobEventId": "b268784cf0a66749f1a05bce259804f5",
        "failureCodes": [],
        "statusChangeReason": []
    }
}
```

EventBridge-Regel zur Nachverfolgung von S3-Batchoperations-Auftragsereignissen

Das folgende Beispiel zeigt, wie Sie eine Regel in Amazon EventBridge erstellen, um von AWS CloudTrail aufgezeichnete S3-Batchoperations-Ereignisse in einem Ziel Ihrer Wahl erfassen.

Dazu erstellen Sie eine Regel, indem Sie alle Schritte unter Erstellen von EventBridge-Regeln, die auf Ereignisse reagieren ausführen. Fügen Sie gegebenenfalls die folgende benutzerdefinierte S3-Batchoperations-Ereignismuster-Richtlinie ein und wählen Sie einen Zielservice aus.

Benutzerdefinierte S3-Batchoperationen-Ereignismusterrichtlinie

```
{
    "source": [
        "aws.s3"
    ],
    "detail-type": [
        "AWS Service Event via CloudTrail"
    ],
    "detail": {
        "eventSource": [
            "s3.amazonaws.com"
        ],
        "eventName": [
            "JobCreated",
            "JobStatusChanged"
        ]
    }
}
```

Die folgenden Beispiele zeigen zwei Batchoperations-Ereignisse, die von einer EventBridge-Ereignisregel an Amazon Simple Queue Service (Amazon SQS) gesendet wurden. Ein Batchoperations-Auftrag durchläuft viele verschiedene Zustände während der Verarbeitung (`New`, `Preparing`, `Active` usw.), sodass Sie davon ausgehen können, dass Sie mehrere Nachrichten für jeden Auftrag erhalten.

Example JobCreated Beispielereignis

```
{
    "version": "0",
    "id": "51dc8145-541c-5518-2349-56d7dffdf2d8",
    "detail-type": "AWS Service Event via CloudTrail",
```

```
    "source": "aws.s3",
    "account": "123456789012",
    "time": "2020-02-27T15:25:49Z",
    "region": "us-east-1",
    "resources": [],
    "detail": {
        "eventVersion": "1.05",
        "userIdentity": {
            "accountId": "11112223334444",
            "invokedBy": "s3.amazonaws.com"
        },
        "eventTime": "2020-02-27T15:25:49Z",
        "eventSource": "s3.amazonaws.com",
        "eventName": "JobCreated",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "s3.amazonaws.com",
        "userAgent": "s3.amazonaws.com",
        "eventID": "7c38220f-f80b-4239-8b78-2ed867b7d3fa",
        "readOnly": false,
        "eventType": "AwsServiceEvent",
        "serviceEventDetails": {
            "jobId": "e849b567-5232-44be-9a0c-40988f14e80c",
            "jobArn": "arn:aws:s3:us-east-1:181572960644:job/
e849b567-5232-44be-9a0c-40988f14e80c",
            "status": "New",
            "jobEventId": "f177ff24f1f097b69768e327038f30ac",
            "failureCodes": [],
            "statusChangeReason": []
        }
    }
}
```

Example JobStatusChanged Auftragsabschluss-Ereignis

```
{
  "version": "0",
  "id": "c8791abf-2af8-c754-0435-fd869ce25233",
  "detail-type": "AWS Service Event via CloudTrail",
  "source": "aws.s3",
  "account": "123456789012",
  "time": "2020-02-27T15:26:42Z",
  "region": "us-east-1",
  "resources": [],
```

```
  "detail": {
    "eventVersion": "1.05",
    "userIdentity": {
      "accountId": "1111222233334444",
      "invokedBy": "s3.amazonaws.com"
    },
    "eventTime": "2020-02-27T15:26:42Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "JobStatusChanged",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "s3.amazonaws.com",
    "userAgent": "s3.amazonaws.com",
    "eventID": "0238c1f7-c2b0-440b-8dbd-1ed5e5833afb",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "serviceEventDetails": {
      "jobId": "e849b567-5232-44be-9a0c-40988f14e80c",
      "jobArn": "arn:aws:s3:us-east-1:181572960644:job/
e849b567-5232-44be-9a0c-40988f14e80c",
      "status": "Complete",
      "jobEventId": "51f5ac17dba408301d56cd1b2c8d1e9e",
      "failureCodes": [],
      "statusChangeReason": []
    }
  }
}
```

Beispiele: Abschlussberichte zu S3-BatchVorgänge

Wenn Sie einen S3-Batchoperations-Auftrag erstellen, können Sie einen Abschlussbericht für alle oder nur für die fehlgeschlagenen Aufträge anfordern. Sofern mindestens eine Aufgabe erfolgreich aufgerufen wurde, generiert S3-Batch-Vorgänge einen Bericht für abgeschlossene, fehlgeschlagene oder abgebrochene Aufträge.

Der Abschlussbericht enthält zusätzliche Informationen zu jeder Aufgabe, darunter den Namen und die Version des Objektschlüssels, Status, Fehlercodes sowie Beschreibungen zu etwaigen Fehlern. Die Beschreibung der Fehler für jede fehlgeschlagene Aufgabe kann herangezogen werden, um Probleme im Rahmen der Auftragserstellung zu diagnostizieren, beispielsweise mit Berechtigungen.

Example Manifest-Ergebnisdatei der höchsten Ebene

Die `manifest.json`-Datei der höchsten Ebene enthält die Position jedes Erfolgsberichts und (sofern Fehler in der Auftragsabwicklung aufgetreten sind) jedes Fehlerberichts (siehe folgendes Beispiel).

```
{
    "Format": "Report_CSV_20180820",
    "ReportCreationDate": "2019-04-05T17:48:39.725Z",
    "Results": [
        {
            "TaskExecutionStatus": "succeeded",
            "Bucket": "my-job-reports",
            "MD5Checksum": "83b1c4cbe93fc893f54053697e10fd6e",
            "Key": "job-f8fb9d89-a3aa-461d-bddc-ea6a1b131955/
results/6217b0fab0de85c408b4be96aeaca9b195a7daa5.csv"
        },
        {
            "TaskExecutionStatus": "failed",
            "Bucket": "my-job-reports",
            "MD5Checksum": "22ee037f3515975f7719699e5c416eaa",
            "Key": "job-f8fb9d89-a3aa-461d-bddc-ea6a1b131955/results/
b2ddad417e94331e9f37b44f1faf8c7ed5873f2e.csv"
        }
    ],
    "ReportSchema": "Bucket, Key, VersionId, TaskStatus, ErrorCode, HTTPStatusCode,
 ResultMessage"
}
```

Example Berichte zu fehlgeschlagenen Aufgaben

Berichte zu fehlgeschlagenen Aufgaben enthalten die folgenden Informationen für alle fehlgeschlagenen Aufgaben:

- `Bucket`
- `Key`
- `VersionId`
- `TaskStatus`
- `ErrorCode`

- `HTTPStatusCode`
- `ResultMessage`

Der folgende Beispielbericht zeigt eine Zeitüberschreitung der AWS Lambda-Funktion, die dazu führt, dass die Anzahl der Fehler den Fehlergrenzwert überschreitet. Deshalb wurde die Kennzeichnung zugewiese `PermanentFailure`.

```
awsexamplebucket1,image_14975,,failed,200,PermanentFailure,"Lambda returned
 function error: {""errorMessage"":""2019-04-05T17:35:21.155Z 2845ca0d-38d9-4c4b-
abcf-379dc749c452 Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_15897,,failed,200,PermanentFailure,"Lambda returned
 function error: {""errorMessage"":""2019-04-05T17:35:29.610Z 2d0a330b-de9b-425f-
b511-29232fde5fe4 Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_14819,,failed,200,PermanentFailure,"Lambda returned function
 error: {""errorMessage"":""2019-04-05T17:35:22.362Z fcf5efde-74d4-4e6d-b37a-
c7f18827f551 Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_15930,,failed,200,PermanentFailure,"Lambda returned function
 error: {""errorMessage"":""2019-04-05T17:35:29.809Z 3dd5b57c-4a4a-48aa-8a35-
cbf027b7957e Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_17644,,failed,200,PermanentFailure,"Lambda
 returned function error: {""errorMessage"":""2019-04-05T17:35:46.025Z
 10a764e4-2b26-4d8c-9056-1e1072b4723f Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_17398,,failed,200,PermanentFailure,"Lambda returned
 function error: {""errorMessage"":""2019-04-05T17:35:44.661Z 1e306352-4c54-4eba-
aee8-4d02f8c0235c Task timed out after 3.00 seconds""}"
```

Example Bericht zu erfolgreichen Aufgaben

Berichte zu erfolgreichen Aufgaben enthalten für alle abgeschlossenen Aufgaben Folgendes:

- `Bucket`
- `Key`
- `VersionId`
- `TaskStatus`
- `ErrorCode`
- `HTTPStatusCode`
- `ResultMessage`

Im folgenden Beispiel hat die Lambda-Funktion das Amazon S3-Objekt in einen anderen Bucket kopiert. Die zurückgegebene Amazon S3-Antwort wird an S3-Batch-Vorgänge zurückgegeben und dann in den endgültigen Abschlussbericht geschrieben.

```
awsexamplebucket1,image_17775,,succeeded,200,,"{u'CopySourceVersionId':
 'xVR78haVKlRnurYofbTfYr3ufYbktF8h', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 39, tzinfo=tzlocal()), u'ETag':
 '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata': {'HTTPStatusCode':
 200, 'RetryAttempts': 0, 'HostId': 'nXNaClIMxEJzWNmeMNQV2KpjbaCJLn0OGoXWZpuVOFS/
iQYWxb3QtTvzX9SVfx2lA3oTKLwImKw=', 'RequestId': '3ED5852152014362', 'HTTPHeaders':
 {'content-length': '234', 'x-amz-id-2': 'nXNaClIMxEJzWNmeMNQV2KpjbaCJLn0OGoXWZpuVOFS/
iQYWxb3QtTvzX9SVfx2lA3oTKLwImKw=', 'x-amz-copy-source-version-id':
 'xVR78haVKlRnurYofbTfYr3ufYbktF8h', 'server': 'AmazonS3', 'x-amz-request-id':
 '3ED5852152014362', 'date': 'Fri, 05 Apr 2019 17:35:39 GMT', 'content-type':
 'application/xml'}}}"
awsexamplebucket1,image_17763,,succeeded,200,,"{u'CopySourceVersionId':
 '6HjOUSim4Wj6BTcbxToXW44pSZ.40pwq', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 39, tzinfo=tzlocal()),
 u'ETag': '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata':
 {'HTTPStatusCode': 200, 'RetryAttempts': 0, 'HostId': 'GiCZNYr8LHd/
Thyk6beTRP96IGZk2sYxujLe13TuuLpq6U2RD3we0YoluuIdm1PRvkMwnEW1aFc=', 'RequestId':
 '1BC9F5B1B95D7000', 'HTTPHeaders': {'content-length': '234', 'x-amz-id-2':
 'GiCZNYr8LHd/Thyk6beTRP96IGZk2sYxujLe13TuuLpq6U2RD3we0YoluuIdm1PRvkMwnEW1aFc=', 'x-
amz-copy-source-version-id': '6HjOUSim4Wj6BTcbxToXW44pSZ.40pwq', 'server': 'AmazonS3',
 'x-amz-request-id': '1BC9F5B1B95D7000', 'date': 'Fri, 05 Apr 2019 17:35:39 GMT',
 'content-type': 'application/xml'}}}"
awsexamplebucket1,image_17860,,succeeded,200,,"{u'CopySourceVersionId':
 'm.MDD0g_QsUnYZ8TBzVFrp.TmjN8PJyX', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 40, tzinfo=tzlocal()), u'ETag':
 '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata': {'HTTPStatusCode':
 200, 'RetryAttempts': 0, 'HostId': 'F9ooZOgpE5g9sNgBZxjdiPHqB4+0DNWgj3qbsir
+sKai4fv7rQEcF2fBN1VeeFc2WH45a9ygb2g=', 'RequestId': '8D9CA56A56813DF3', 'HTTPHeaders':
 {'content-length': '234', 'x-amz-id-2': 'F9ooZOgpE5g9sNgBZxjdiPHqB4+0DNWgj3qbsir
+sKai4fv7rQEcF2fBN1VeeFc2WH45a9ygb2g=', 'x-amz-copy-source-version-id':
 'm.MDD0g_QsUnYZ8TBzVFrp.TmjN8PJyX', 'server': 'AmazonS3', 'x-amz-request-id':
 '8D9CA56A56813DF3', 'date': 'Fri, 05 Apr 2019 17:35:40 GMT', 'content-type':
 'application/xml'}}}"
```

Steuern von Zugriffs- und Labeling-Aufträgen mithilfe von Markierungen

Sie können den Zugriff auf Ihre S3-Batch-Vorgänge-Aufträge kennzeichnen und steuern, indem Sie Markierungen hinzufügen. Markierungen können verwendet werden, um zu ermitteln, wer für

einen Batchoperations-Auftrag verantwortlich ist. Das Vorhandensein von Aufgabe-Markierungen kann einem Benutzer die Möglichkeit gewähren oder einschränken, eine Aufgabe abzubrechen, eine Aufgabe im Bestätigungsstatus zu aktivieren oder die Prioritätsstufe einer Aufgabe zu ändern. Sie können Aufträge mit ihnen zugeordneten Markierungen erstellen oder Sie können Aufträgen nach ihrer Erstellung Markierungen hinzufügen. Jedes Tag ist ein Schlüssel-Wert-Paar, das beim Erstellen des Auftrags oder später aktualisiert werden kann.

> **Warning**
>
> Auftrags-Markierungen sollten keine vertraulichen Informationen oder persönlichen Daten enthalten.

Betrachten Sie das folgende Tagging-Beispiel: Angenommen, Ihre Finanzabteilung soll einen Batchoperations-Auftrag erstellen. Sie können eine AWS Identity and Access Management (IAM) Richtlinie schreiben, die es einem Benutzer ermöglicht `CreateJob` aufzurufen, vorausgesetzt, der Auftrag wird mit dem `Department`-Tag erstellt, der den zugewiesenen Wert `Finance` hat. Außerdem können Sie diese Richtlinie an alle Benutzer anfügen, die Mitglieder der Finanzabteilung sind.

Wenn Sie mit diesem Beispiel fortfahren, können Sie eine Richtlinie schreiben, die es einem Benutzer ermöglicht, die Priorität aller Aufträge, die über die gewünschten Markierungen verfügen, zu aktualisieren oder alle Aufträge, die diese Markierungen besitzen, abzubrechen. Weitere Informationen finden Sie unter the section called “Steuern von Berechtigungen”.

Sie können Markierungen beim Erstellen neuen S3-Batchoperations-Aufträgen oder bereits vorhandenen Aufträgen hinzufügen.

Beachten Sie die folgenden Tag-Einschränkungen:

- Sie können einem Auftrag bis zu 50 Markierungen zuordnen, solange diese über eindeutige Tag-Schlüssel verfügen.
- Ein Tag-Schlüssel kann maximal 128 Unicode-Zeichen lang sein, und die Tag-Werte können bis zu 256 Unicode-Zeichen lang sein.
- Bei Schlüsseln und Werten wird die Groß-/Kleinschreibung berücksichtigt.

Weitere Informationen zu Tag-Einschränkungen finden Sie unter Einschränkungen benutzerdefinierter Markierungen im AWS Billing and Cost Management Benutzerhandbuch.

API-Vorgänge im Zusammenhang mit S3-Batchoperations-Auftragsmarkierung

Amazon S3 unterstützt die folgenden API-Vorgänge, die spezifisch für das S3-Batchoperations-Auftrag-Tagging sind:

- GetJobTagging – Gibt den Markierungenatz zurück, der mit einem Batchoperations-Auftrag verknüpft ist.
- PutJobTagging – Ersetzt den Markierungenatz, der einem Auftrag zugeordnet ist. Es gibt zwei unterschiedliche Szenarien der S3-Batchoperations-Auftrags-Tag-Verwaltung unter Verwendung dieser API-Aktion:
  - Auftrag ohne Markierungen – Sie können einen Satz von Markierungen zu einem Auftrag hinzufügen (der Auftrag hat keine vorherigen Markierungen).
  - Auftrag verfügt über einen Satz vorhandener Markierungen – Um den vorhandenen Tag-Satz zu ändern, können Sie entweder den vorhandenen Tag-Satz vollständig ersetzen oder Änderungen innerhalb des vorhandenen Tag-Satzes vornehmen, indem Sie den vorhandenen Tag-Satz mit GetJobTagging abrufen, diesen Tag-Satz ändern und mit dieser API-Aktion diesen Tag-Satz gegen einen von Ihnen modifizierten auswechseln.

    > **Note**
    >
    > Wenn Sie diese Anforderung mit einem leeren Tag-Satz senden, löscht S3-Batch-Vorgänge den vorhandenen Tag-Satz für das Objekt. Wenn Sie diese Methode verwenden, wird Ihnen eine Tier 1-Anforderung (PUT) berechnet. Weitere Informationen finden Sie unter Amazon S3 – Preise.
    >
    > Wenn Sie vorhandene Markierungen für Ihren Batchoperations-Auftrag löschen möchten, wird die Aktion `DeleteJobTagging` bevorzugt, da sie dasselbe Ergebnis ohne anfallende Gebühren erzielt.

- DeleteJobTagging – Löscht den Markierungenatz, der mit einem Batchoperations-Auftrag verknüpft ist.

Erstellen eines Batchoperations-Auftrags mit Auftrags-Markierungen zur Kennzeichnung

Sie können den Zugriff auf Ihre S3-Batch-Vorgänge-Aufträge kennzeichnen und steuern, indem Sie Markierungen hinzufügen. Markierungen können verwendet werden, um zu ermitteln, wer für einen Batchoperations-Auftrag verantwortlich ist. Sie können Aufträge mit ihnen zugeordneten

Markierungen erstellen oder Sie können Aufträgen nach ihrer Erstellung Markierungen hinzufügen. Weitere Informationen finden Sie unter the section called “Verwenden von Markierungen”.

Verwendung der AWS CLI

Im folgenden AWS CLI-CLI-Beispiel wird der S3-Batchoperations-Auftrag `S3PutObjectCopy` mit Auftrags-Markierungen als Bezeichnungen für den Auftrag erstellt.

1. Wählen Sie die Aktion oder `OPERATION` aus, die der Batchoperations-Auftrag ausführen soll, und wählen Sie Ihre `TargetResource`.

```
read -d '' OPERATION <<EOF
{
  "S3PutObjectCopy": {
    "TargetResource": "arn:aws:s3:::destination-bucket"
  }
}
EOF
```

2. Identifizieren Sie die Auftrags-`TAGS`, die Sie für den Auftrag wünschen. In diesem Fall wenden Sie beiden Markierungen `department` und `FiscalYear` mit den Werten `Marketing` bzw. `2020` an.

```
read -d '' TAGS <<EOF
[
  {
    "Key": "department",
    "Value": "Marketing"
  },
  {
    "Key": "FiscalYear",
    "Value": "2020"
  }
]
EOF
```

3. Geben Sie das `MANIFEST` für den Batchoperations-Auftrag an.

```
read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "EXAMPLE_S3BatchOperations_CSV_20180820",
```

```
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::example-bucket/example_manifest.csv",
    "ETag": "example-5dc7a8bfb90808fc5d546218"
  }
}
EOF
```

4. Konfigurieren Sie den REPORT für den Batchoperations-Auftrag.

```
read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::example-report-bucket",
  "Format": "Example_Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/copy-with-replace-metadata",
  "ReportScope": "AllTasks"
}
EOF
```

5. Führen Sie die Aktion `create-job` aus, um Ihren Batchoperations-Auftrag mit Eingaben zu erstellen, die in den vorherigen Schritten festgelegt wurden.

```
aws \
    s3control create-job \
    --account-id 123456789012 \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn arn:aws:iam::123456789012:role/batch-operations-role \
    --tags "${TAGS//$'\n'/}" \
    --client-request-token "$(uuidgen)" \
    --region us-west-2 \
    --description "Copy with Replace Metadata";
```

Verwenden des AWS-SDK für Java

Example

Im folgenden Beispiel wird ein S3-Batchoperations-Auftrag mit Markierungen unter Verwendung des erstellt AWS SDK for Java.

```
public String createJob(final AWSS3ControlClient awss3ControlClient) {
    final String manifestObjectArn = "arn:aws:s3:::example-manifest-bucket/
manifests/10_manifest.csv";
    final String manifestObjectVersionId = "example-5dc7a8bfb90808fc5d546218";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new
 JobManifestSpec().withFormat(JobManifestFormat.S3InventoryReport_CSV_20161130);

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::example-report-bucket";
    final String jobReportPrefix = "example-job-reports";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final String lambdaFunctionArn = "arn:aws:lambda:us-
west-2:123456789012:function:example-function";

    final JobOperation jobOperation = new JobOperation()
            .withLambdaInvoke(new
 LambdaInvokeOperation().withFunctionArn(lambdaFunctionArn));

    final S3Tag departmentTag = new
 S3Tag().withKey("department").withValue("Marketing");
    final S3Tag fiscalYearTag = new S3Tag().withKey("FiscalYear").withValue("2020");
```

```
    final String roleArn = "arn:aws:iam::123456789012:role/example-batch-operations-
role";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Test lambda job")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withTags(departmentTag, fiscalYearTag)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Löschen der Markierungen aus einem S3-Batchoperations-Auftrag

Sie können diese Beispiele verwenden, um die Markierungen aus einem Batchoperations-Auftrag zu löschen.

Verwendung der AWS CLI

Im folgenden Beispiel werden die Markierungen aus einem Batchoperations-Auftrag unter Verwendung der gelöscht AWS CLI.

```
aws \
    s3control delete-job-tagging \
    --account-id 123456789012 \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

Löschen der Auftrags-Markierungen eines Batchoperations-Auftrags

Example

Im folgenden Beispiel werden die Markierungen eines S3-Batchoperations-Auftrags unter Verwendung des -SDK für Java gelöscht AWS SDK for Java.

```
public void deleteJobTagging(final AWSS3ControlClient awss3ControlClient,
                             final String jobId) {
    final DeleteJobTaggingRequest deleteJobTaggingRequest = new
 DeleteJobTaggingRequest()
            .withJobId(jobId);

    final DeleteJobTaggingResult deleteJobTaggingResult =
                awss3ControlClient.deleteJobTagging(deleteJobTaggingRequest);
}
```

Einfügen von Auftrags-Markierungen für einen bestehenden S3-Batchoperations-Auftrag

Sie können PutJobTagging verwenden, um Auftrags-Markierungen zu Ihren bestehenden S3-Batchoperationsaufträge hinzuzufügen. Weitere Informationen finden Sie in den folgenden Beispielen.

Verwendung der AWS CLI

Im Folgenden finden Sie ein Beispiel für `s3control put-job-tagging` das Hinzufügen von Auftrags-Markierungen zu Ihrem S3-Batchoperations-Auftrag unter der Verwendung derAWS CLI-CLI.

> **Note**
>
> Wenn Sie diese Anforderung mit einem leeren Tag-Satz senden, löscht S3-Batch-Vorgänge den vorhandenen Tag-Satz für das Objekt. Wenn Sie diese Methode verwenden, wird Ihnen eine Tier 1-Anforderung (PUT) berechnet. Weitere Informationen finden Sie unter Amazon S3 – Preise.
>
> Wenn Sie vorhandene Markierungen für Ihren Batchoperations-Auftrag löschen möchten, wird die Aktion `DeleteJobTagging` bevorzugt, da sie dasselbe Ergebnis ohne anfallende Gebühren erzielt.

1. Identifizieren Sie die Auftrags-TAGS, die Sie für den Auftrag wünschen. In diesem Fall wenden Sie beiden Markierungen `department` und `FiscalYear` mit den Werten `Marketing` bzw. `2020` an.

```
read -d '' TAGS <<EOF
[
  {
    "Key": "department",
    "Value": "Marketing"
  },
  {
    "Key": "FiscalYear",
    "Value": "2020"
  }
]
EOF
```

2. Führen Sie die `put-job-tagging`-Aktion mit den erforderlichen Parametern aus.

```
aws \
    s3control put-job-tagging \
    --account-id 123456789012 \
    --tags "${TAGS//$'\n'/}" \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

Verwenden des AWS-SDK für Java

Example

Im folgenden Beispiel werden die Markierungen eines S3-Batchoperations-Auftrags unter Verwendung des -SDK für Java eingefügt AWS SDK for Java.

```
public void putJobTagging(final AWSS3ControlClient awss3ControlClient,
                          final String jobId) {
    final S3Tag departmentTag = new
 S3Tag().withKey("department").withValue("Marketing");
    final S3Tag fiscalYearTag = new S3Tag().withKey("FiscalYear").withValue("2020");

    final PutJobTaggingRequest putJobTaggingRequest = new PutJobTaggingRequest()
            .withJobId(jobId)
```

```
            .withTags(departmentTag, fiscalYearTag);

    final PutJobTaggingResult putJobTaggingResult =
 awss3ControlClient.putJobTagging(putJobTaggingRequest);
}
```

Abrufen der Markierungen eines S3-Batchoperations-Auftrags

Sie können `GetJobTagging` verwenden, um die Markierungen eines S3-Batchoperations-Auftrags zurückzugeben. Weitere Informationen finden Sie in den folgenden Beispielen.

Verwendung der AWS CLI

Im folgenden Beispiel werden die Markierungen eines Batchoperations-Auftrags unter Verwendung der CLI abgerufen AWS CLI.

```
aws \
    s3control get-job-tagging \
    --account-id 123456789012 \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

Verwenden des AWS-SDK für Java

Example

Im folgenden Beispiel werden die Markierungen eines S3 Batchoperations-Auftrags unter Verwendung der CLI abgerufen AWS SDK for Java.

```
public List<S3Tag> getJobTagging(final AWSS3ControlClient awss3ControlClient,
                                 final String jobId) {
    final GetJobTaggingRequest getJobTaggingRequest = new GetJobTaggingRequest()
            .withJobId(jobId);

    final GetJobTaggingResult getJobTaggingResult =
            awss3ControlClient.getJobTagging(getJobTaggingRequest);

    final List<S3Tag> tags = getJobTaggingResult.getTags();

    return tags;
}
```

Steuern von Berechtigungen für S3-Batch-Vorgänge mithilfe von Auftrags-Markierungen

Zur Unterstützung bei der Verwaltung Ihrer S3-Batch-Vorgänge-Aufträge können Sie Auftrags-Markierungen hinzufügen. Mit Auftrags-Markierungen können Sie den Zugriff auf Ihre BatchVorgängeaufträge steuern und erzwingen, dass Markierungen angewendet werden, wenn ein Auftrag erstellt wird.

Sie können bis zu 50 Auftrag-Markierungen auf jeden Batchoperations-Auftrag anwenden. Auf diese Weise können Sie sehr detaillierte Richtlinien festlegen, die die Gruppe von Benutzern einschränken, die den Auftrag bearbeiten können. Aufgaben-Markierungen können einem Benutzer die Möglichkeit gewähren oder einschränken, eine Aufgabe abzubrechen, eine Aufgabe im Bestätigungsstatus zu aktivieren oder die Prioritätsstufe einer Aufgabe zu ändern. Darüber hinaus können Sie erzwingen, dass Markierungen auf alle neuen Aufträge angewendet werden, und die zulässigen Schlüssel-Wert-Paare für die Markierungen angeben. Sie können alle diese Bedingungen mit derselben IAM-Richtliniensprache ausdrücken. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 in der Service-Autorisierungs-Referenz.

Das folgende Beispiel zeigt, wie Sie S3-Batchoperations-Auftrags-Markierungen verwenden können, um Benutzern die Berechtigung zu erteilen, nur die Aufträge zu erstellen und zu bearbeiten, die in einer bestimmten Abteilung (z. B. der Abteilung Finanzen oder Compliance) ausgeführt werden. Sie können Aufträge auch basierend auf der Entwicklungsphase zuweisen, auf die sie sich beziehen, z. B QS oder Produktion.

In diesem Beispiel verwenden Sie S3-Batchoperationenauftrags-Tags in AWS Identity and Access Management (IAM)-Richtlinien, um Benutzern die Berechtigung zu erteilen, nur die Aufträge zu erstellen und zu bearbeiten, die in ihrer Abteilung ausgeführt werden. Sie weisen Aufträge basierend auf der Entwicklungsphase zu, mit der sie verbunden sind, wie QS oder Produktion.

In diesem Beispiel werden die folgenden Abteilungen verwendet, die Batch-Vorgänge auf jeweils unterschiedliche Weise einsetzen:

- Finanzen
- Compliance
- Business Intelligence
- Entwicklung

Themen

Steuern des Zugriffs durch Zuweisen von Markierungen zu Benutzern und Ressourcen

In diesem Szenario verwenden die Administratoren attributbasierte Zugriffssteuerung (Attribute-based Access Control, ABAC). ABAC ist eine IAM-Autorisierungsstrategie, die Berechtigungen definiert, indem Tags sowohl an Benutzer als auch an AWS Ressourcen angehängt werden.

Benutzern und Jobs wird eines der folgenden Abteilungstags zugewiesen:

Schlüssel : Wert

- `department : Finance`
- `department : Compliance`
- `department : BusinessIntelligence`
- `department : Engineering`

> **Note**
>
> Bei Tag-Schlüsseln und -Werten muss die Groß-/Kleinschreibung beachtet werden.

Mit der ABAC-Zugriffssteuerungsstrategie erteilen Sie einem Benutzer in der Finanzabteilung die Berechtigung, S3-Batch-Vorgänge-Aufträge innerhalb seiner Abteilung zu erstellen und zu verwalten, indem Sie das Tag `department=Finance` mit dem Benutzer verknüpfen.

Darüber hinaus können Sie dem IAM-Benutzer eine verwaltete Richtlinie anfügen, die es jedem Benutzer in seinem Unternehmen ermöglicht, S3-BatchVorgängeaufträge innerhalb seiner jeweiligen Abteilungen zu erstellen oder zu ändern.

Die Richtlinie in diesem Beispiel enthält drei Richtlinienanweisungen:

- Die erste Anweisung in der Richtlinie ermöglicht es dem Benutzer, einen Batchoperations-Auftrag zu erstellen, vorausgesetzt, die Auftragserstellungsanforderung enthält ein Auftrags-Tag, das

der jeweiligen Abteilung entspricht. Dies wird mithilfe der Syntax `"${aws:PrincipalTag/department}"` ausgedrückt, die zum Zeitpunkt der Richtlinienauswertung durch das Abteilungs-Tag des Benutzers ersetzt wird. Die Bedingung ist erfüllt, wenn der für das Abteilungs-Tag in der Anforderung (`"aws:RequestTag/department"`) angegebene Wert mit dem der Abteilung des Benutzers übereinstimmt.

- Die zweite Anweisung in der Richtlinie ermöglicht es Benutzern, die Priorität von Aufträgen zu ändern oder den Status eines Auftrags zu aktualisieren, sofern der Auftrag, den der Benutzer aktualisiert, mit der Abteilung des Benutzers übereinstimmt.
- Die dritte Anweisung ermöglicht es einem Benutzer, die Markierungen eines Batchoperations-Auftrags jederzeit über eine `PutJobTagging`-Anforderung zu aktualisieren, solange (1) das Abteilung-Tag erhalten bleibt und (2) sich der Auftrag, der aktualisiert wird, innerhalb der Abteilung des Benutzers befindet.

```
{
      "Version": "2012-10-17",
      "Statement": [
            {
                  "Effect": "Allow",
                  "Action": "s3:CreateJob",
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:RequestTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        },
            {
                  "Effect": "Allow",
                  "Action": [
                        "s3:UpdateJobPriority",
                        "s3:UpdateJobStatus"
            ],
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                }
            }
```

```
        },
            {
                  "Effect": "Allow",
                  "Action": "s3:PutJobTagging",
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:RequestTag/department": "${aws:PrincipalTag/
department}",
                              "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        }
    ]
}
```

Markieren von BatchVorgängeaufträgen nach Stufe und Durchsetzen von Limits für die Auftragspriorität

Alle S3-BatchVorgängeaufträge haben eine numerische Priorität, anhand derer Amazon S3 entscheidet, in welcher Reihenfolge die Aufträge ausgeführt werden sollen. In diesem Beispiel beschränken Sie die maximale Priorität, die die meisten Benutzer Aufträgen zuweisen können, wobei höhere Prioritätsbereiche für eine begrenzte Gruppe von berechtigten Benutzern reserviert sind:

- Prioritätsbereich der QA-Stufe (niedrig): 1-100
- Prioritätsbereich der Produktionsstufe (hoch): 1-300

Dazu führen Sie einen neuen Tag-Satz ein, der die Phase des Auftrags repräsentiert:

Schlüssel : Wert

- `stage : QA`
- `stage : Production`

Erstellen und Aktualisieren von Aufträgen mit geringer Priorität in einer Abteilung

Mit dieser Richtlinie werden neben der abteilungsbasierten Einschränkung zwei neue Einschränkungen für die Erstellung und Aktualisierung von S3-BatchVorgängeaufträgen eingeführt:

- Es ermöglicht Benutzern, Aufträge in ihrer Abteilung mit einer neuen Bedingung zu erstellen oder zu aktualisieren, die erfordert, dass der Auftrag das Tag enthäl `stage=QA`.
- Es ermöglicht Benutzern, Aufträge mit einer maximalen Priorität von bis 100 zu erstellen oder zu aktualisieren.

```
{
        "Version": "2012-10-17",
        "Statement": [
        {
        "Effect": "Allow",
        "Action": "s3:CreateJob",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/department": "${aws:PrincipalTag/department}",
                "aws:RequestTag/stage": "QA"
            },
            "NumericLessThanEquals": {
                "s3:RequestJobPriority": 100
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:UpdateJobStatus"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:UpdateJobPriority",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}",
                "aws:ResourceTag/stage": "QA"
```

```
            },
            "NumericLessThanEquals": {
                "s3:RequestJobPriority": 100
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:PutJobTagging",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/department" : "${aws:PrincipalTag/department}",
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}",
                "aws:RequestTag/stage": "QA",
                "aws:ResourceTag/stage": "QA"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:GetJobTagging",
        "Resource": "*"
    }
    ]
}
```

Erstellen und Aktualisieren von Aufträgen mit hoher Priorität in einer Abteilung

Einer kleinen Anzahl von Benutzern muss u. U. möglich sein, Aufträge hoher Priorität in QS oder Produktion zu erstellen. Um diesen Bedarf zu unterstützen, erstellen Sie eine verwaltete Richtlinie durch Abwandlung einer Richtlinie mit niedriger Priorität im vorherigen Abschnitt.

Diese Richtlinie gewährt die folgenden Aktionen:

- Ermöglicht Benutzern, Aufträge in ihrer Abteilung entweder mit dem Tag `stage=QA` oder `stage=Production` zu erstellen oder zu aktualisieren.
- Ermöglicht Benutzern, bei der Erstellung oder Aktualisierung eines Auftrags als Priorität bis zu 300 anzugeben.

```
{
```

```
      "Version": "2012-10-17",
      "Statement": [
          {
                "Effect": "Allow",
                "Action": "s3:CreateJob",
                "Resource": "*",
                "Condition": {
                      "ForAnyValue:StringEquals": {
                            "aws:RequestTag/stage": [
                                  "QA",
                                  "Production"
                    ]
                },
                      "StringEquals": {
                            "aws:RequestTag/department": "${aws:PrincipalTag/
department}"
                },
                      "NumericLessThanEquals": {
                            "s3:RequestJobPriority": 300
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": [
                      "s3:UpdateJobStatus"
            ],
                "Resource": "*",
                "Condition": {
                      "StringEquals": {
                            "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": "s3:UpdateJobPriority",
                "Resource": "*",
                "Condition": {
                      "ForAnyValue:StringEquals": {
                            "aws:ResourceTag/stage": [
                                  "QA",
                                  "Production"
```

```
                    ]
                },
                      "StringEquals": {
                            "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                },
                      "NumericLessThanEquals": {
                            "s3:RequestJobPriority": 300
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": "s3:PutJobTagging",
                "Resource": "*",
                "Condition": {
                      "StringEquals": {
                            "aws:RequestTag/department": "${aws:PrincipalTag/
department}",
                            "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                },
                      "ForAnyValue:StringEquals": {
                            "aws:RequestTag/stage": [
                                  "QA",
                                  "Production"
                    ],
                            "aws:ResourceTag/stage": [
                                  "QA",
                                  "Production"
                    ]
                }
            }
        }
    ]
}
```

Verwalten der S3-Objektsperre mit S3-BatchVorgänge

Mit der S3-Objektsperre können Sie eine rechtliche Aufbewahrungsfrist für eine Objektversion festlegen. Wie das Festlegen von Aufbewahrungszeiträumen verhindern auch rechtliche Aufbewahrungsfristen das Überschreiben oder Löschen von Objektversionen. Mit rechtlichen Aufbewahrungsfristen sind jedoch keine Aufbewahrungszeiträume verknüpft. Sie sind gültig, bis

sie entfernt werden. Weitere Informationen finden Sie unter S3-Objektsperre aufgrund gesetzlicher Aufbewahrungsfristen.

Informationen zur Verwendung von S3-Batch-Vorgängen mit Object Lock zum Hinzufügen von rechtlichen Holds zu vielen Amazon S3-Objekten gleichzeitig finden Sie in den folgenden Abschnitten.

Themen

Aktivieren der S3-Objektsperre mit S3-BatchVorgänge

Sie können S3-Batch-Vorgänge mit der S3-Objektsperre verwenden, um für viele Amazon S3-Objekte gleichzeitig die Aufbewahrung zu verwalten oder eine rechtliche Aufbewahrungsfrist zu wählen. Sie geben die Liste der Zielobjekte in Ihrem Manifest an und senden sie zur Fertigstellung an BatchVorgänge. Weitere Informationen erhalten Sie unter the section called “Aufrechterhaltung der Objektsperre” und the section called “Objektsperre aufgrund gesetzlicher Aufbewahrungsfristen”.

In den folgenden Beispielen wird gezeigt, wie Sie eine IAM-Rolle mit Berechtigungen für S3-Batch-Vorgänge erstellen und die Rollenberechtigungen aktualisieren, um Aufträge zu erstellen, die die Objektsperre aktivieren. Ersetzen Sie in den Beispielen alle Variablenwerte durch diejenigen, die Ihren Anforderungen entsprechen. Sie müssen auch über ein CSV-Manifest verfügen, das die Objekte für Ihren S3-Batchoperations-Auftrag identifiziert. Weitere Informationen finden Sie unter the section called “Angeben eines Manifests”.

Verwendung der AWS CLI

1. Erstellen Sie eine IAM-Rolle und weisen Sie zur Ausführung Berechtigungen für S3-Batch-Vorgänge zu.

   Dieser Schritt ist für alle S3-BatchVorgängeaufträge erforderlich.

```
export AWS_PROFILE='aws-user'

read -d '' bops_trust_policy <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "batchoperations.s3.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
aws iam create-role --role-name bops-objectlock --assume-role-policy-document
 "${bops_trust_policy}"
```

2. Richten Sie S3-Batch-Vorgänge zur Ausführung mit S3-Objektsperre ein.

   In diesem Schritt lassen Sie der Rolle Folgendes zu:

   a. Führen Sie die Objektsperre in dem S3-Bucket aus, der die Zielobjekte enthält, für die Batch-Vorgänge ausgeführt werden soll.

   b. Lesen des S3-Buckets, in dem sich die Manifest-CSV-Datei und die Objekte befinden.

   c. Schreiben Sie die Ergebnisse des S3-Batchoperations-Auftrags in den Berichts-Bucket.

```
read -d '' bops_permissions <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetBucketObjectLockConfiguration",
            "Resource": [
                "arn:aws:s3:::{{ManifestBucket}}"
            ]
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::{{ManifestBucket}}/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::{{ReportBucket}}/*"
            ]
        }
    ]
}
EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name object-lock-
permissions --policy-document "${bops_permissions}"
```

Verwenden des AWS-SDK für Java

In den folgenden Beispielen wird gezeigt, wie Sie eine IAM-Rolle mit Berechtigungen für S3-Batch-Vorgänge erstellen und die Rollenberechtigungen aktualisieren, um Aufträge zu erstellen, die die Objektsperre mithilfe des aktivieren AWS SDK for Java. Ersetzen Sie im Code alle Variablenwerte durch diejenigen, die Ihren Anforderungen entsprechen. Sie müssen auch über ein CSV-Manifest verfügen, das die Objekte für Ihren S3-Batchoperations-Auftrag identifiziert. Weitere Informationen finden Sie unter the section called “Angeben eines Manifests”.

Führen Sie die folgenden Schritte aus:

1. Erstellen Sie eine IAM-Rolle und weisen Sie zur Ausführung Berechtigungen für S3-Batch-Vorgänge zu. Dieser Schritt ist für alle S3-BatchVorgängeaufträge erforderlich.

2. Richten Sie S3-Batch-Vorgänge zur Ausführung mit S3-Objektsperre ein.

   Sie lassen der Rolle Folgendes zu:

   1. Führen Sie die Objektsperre in dem S3-Bucket aus, der die Zielobjekte enthält, für die Batch-Vorgänge ausgeführt werden soll.
   2. Lesen des S3-Buckets, in dem sich die Manifest-CSV-Datei und die Objekte befinden.
   3. Schreiben Sie die Ergebnisse des S3-Batchoperations-Auftrags in den Berichts-Bucket.

```
public void createObjectLockRole() {
    final String roleName = "bops-object-lock";

    final String trustPolicy = "{" +
            "  \"Version\": \"2012-10-17\", " +
            "  \"Statement\": [ " +
            "    { " +
            "      \"Effect\": \"Allow\", " +
            "      \"Principal\": { " +
            "        \"Service\": [" +
            "          \"batchoperations.s3.amazonaws.com\"" +
            "        ]" +
            "      }, " +
            "      \"Action\": \"sts:AssumeRole\" " +
            "    } " +
            "  ]" +
            "}";

    final String bopsPermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": \"s3:GetBucketObjectLockConfiguration\"," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket\"" +
            "            ]" +
            "        }," +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:GetObject\"," +
```

```
            "                \"s3:GetObjectVersion\"," +
            "                \"s3:GetBucketLocation\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket/*\"" +
            "            ]" +
            "        }," +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:PutObject\"," +
            "                \"s3:GetBucketLocation\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ReportBucket/*\"" +
            "            ]" +
            "        }" +
            "    ]" +
            "}";

    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final CreateRoleRequest createRoleRequest = new CreateRoleRequest()
            .withAssumeRolePolicyDocument(bopsPermissions)
            .withRoleName(roleName);

    final CreateRoleResult createRoleResult = iam.createRole(createRoleRequest);

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withPolicyDocument(bopsPermissions)
            .withPolicyName("bops-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
```

Festlegen der Objektsperrenaufbewahrung mit BatchVorgänge

Im folgenden Beispiel kann die Regel die S3-Objektsperrenaufbewahrung für Ihre Objekte im Manifest-Bucket festlegen.

Sie aktualisieren die Rolle, sodass sie `s3:PutObjectRetention`-Berechtigungen einschließt und Sie die Objektsperrenaufbewahrung für die Objekte in Ihrem Bucket ausführen können.

Verwendung der AWS CLI

```
export AWS_PROFILE='aws-user'

read -d '' retention_permissions <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectRetention"
            ],
            "Resource": [
                "arn:aws:s3:::{{ManifestBucket}}/*"
            ]
        }
    ]
}
EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name retention-permissions
 --policy-document "${retention_permissions}"
```

Verwenden des AWS-SDK für Java

```
public void allowPutObjectRetention() {
    final String roleName = "bops-object-lock";

    final String retentionPermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:PutObjectRetention\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket*\"" +
            "            ]" +
```

```
            "        }" +
            "    ]" +
            "}";

    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withPolicyDocument(retentionPermissions)
            .withPolicyName("retention-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
```

Verwendung von S3-Batch-Vorgänge mit dem Compliance-Modus der S3-Objektsperrenaufbewahrung

Das folgende Beispiel baut auf den vorherigen Beispielen zum Erstellen einer Vertrauensrichtlinie sowie zum Festlegen von S3-Batchoperations- und S3-Objektsperren-Konfigurations-Berechtigungen für Ihre Objekte auf. In diesem Beispiel wird der Aufbewahrungsmodus auf `COMPLIANCE` und das `retain until date` auf 1. Januar 2020 festgelegt. Es erstellt einen Auftrag, der auf Objekte im Manifest-Bucket abzielt und die Ergebnisse im Berichts-Bucket meldet, den Sie identifiziert haben.

Verwendung der AWS CLI

Example Festlegen der Compliance von Erwähnungen für mehrere Objekte

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "Retention": {
      "RetainUntilDate":"2025-01-01T00:00:00",
      "Mode":"COMPLIANCE"
    }
  }
```

```
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/compliance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucket",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/compliance-objects-bops",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Set compliance retain-until to 1 Jul 2030";
```

Example Erweitern Sie das **COMPLIANCE** des **retain until date**-Modus bis zum 15. Januar 2020

Im folgenden Beispiel wird das COMPLIANCE des retain until date-Modus bis zum 15. Januar 2025 erweitert.

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "Retention": {
      "RetainUntilDate":"2025-01-15T00:00:00",
      "Mode":"COMPLIANCE"
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/compliance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucket",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
```

```
  "Prefix": "reports/compliance-objects-bops",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Extend compliance retention to 15 Jan 2020";
```

Verwenden des AWS-SDK für Java

Example Stellen Sie den Aufbewahrungsmodus auf COMPLIANCE und das Aufbewahren-bis-Datum auf den 1. Januar 2020 ein.

```
public String createComplianceRetentionJob(final AWSS3ControlClient awss3ControlClient)
 throws ParseException {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/compliance-objects-
manifest.csv";
    final String manifestObjectVersionId = "your-object-version-Id";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/compliance-objects-bops";
```

```
    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
    final Date janFirst = format.parse("01/01/2020");

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()
                            .withMode(S3ObjectLockRetentionMode.COMPLIANCE)
                            .withRetainUntilDate(janFirst)));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Set compliance retain-until to 1 Jan 2020")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Example Erweitern des **`COMPLIANCE`** des **`retain until date`**-Modus

Im folgenden Beispiel wird das `COMPLIANCE` des `retain until date`-Modus bis zum 15. Januar 2020 erweitert.

```
public String createExtendComplianceRetentionJob(final AWSS3ControlClient
 awss3ControlClient) throws ParseException {
```

```
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/compliance-objects-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/compliance-objects-bops";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
    final Date jan15th = format.parse("15/01/2020");

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()
                            .withMode(S3ObjectLockRetentionMode.COMPLIANCE)
                            .withRetainUntilDate(jan15th)));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Extend compliance retention to 15 Jan 2020")
            .withManifest(manifestToPublicApi)
```

```
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Verwenden von S3-Batch-Vorgänge mit dem Governance-Modus der S3-Objektsperrenaufbewahrung

Das folgende Beispiel baut auf dem vorherigen Beispiel zum Erstellen einer Vertrauensrichtlinie sowie zum Festlegen von S3-Batchoperations- und S3-Objektsperren-Konfigurations-Berechtigungen auf. Es zeigt, wie die S3-Objektsperrenaufbewahrungs-Governance mit dem `retain until date` 30. Januar 2025 auf mehrere Objekte angewendet wird. Es erstellt einen Batchoperations-Auftrag, der den Manifest-Bucket verwendet und die Ergebnisse im Berichts-Bucket meldet.

Verwendung der AWS CLI

Example Wenden Sie die S3-Objektsperrenaufbewahrungs-Governance auf mehrere Objekte mit dem Aufbewahren-bis-Datum bis zum 30. Januar 2020 an

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "Retention": {
      "RetainUntilDate":"2025-01-30T00:00:00",
      "Mode":"GOVERNANCE"
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
```

```
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/governance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucketT",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/governance-objects",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Put governance retention";
```

Example Umgehen Sie Aufbewahrungs-Governance über mehrere Objekte

Das folgende Beispiel baut auf dem vorherigen Beispiel zum Erstellen einer Vertrauensrichtlinie sowie zum Festlegen von S3-Batchoperations- und S3-Objektsperren-Konfigurations-Berechtigungen auf. Es zeigt, wie Sie die Aufbewahrungs-Governance über mehrere Objekte hinweg umgehen und

erstellt einen Batchoperations-Auftrag, der den Manifest-Bucket verwendet und die Ergebnisse im Berichts-Bucket meldet.

```
export AWS_PROFILE='aws-user'

read -d '' bypass_governance_permissions <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:BypassGovernanceRetention"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        }
    ]
}
EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name bypass-governance-
permissions --policy-document "${bypass_governance_permissions}"

export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "BypassGovernanceRetention": true,
    "Retention": {
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
```

```
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/governance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::REPORT_BUCKET",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/bops-governance",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Remove governance retention";
```

Verwenden des AWS-SDK für Java

Das folgende Beispiel baut auf dem vorherigen Beispiel zum Erstellen einer Vertrauensrichtlinie sowie zum Festlegen von S3-Batchoperations- und S3-Objektsperren-Konfigurations-Berechtigungen auf. Es zeigt, wie die S3-Objektsperrenaufbewahrungs-Governance mit dem auf den 30. Januar 2020 festgelegten `retain until date` auf mehrere Objekte angewendet wird. Es erstellt einen Batchoperations-Auftrag, der den Manifest-Bucket verwendet und die Ergebnisse im Berichts-Bucket meldet.

Example Wenden Sie die S3-Objektsperrenaufbewahrungs-Governance auf mehrere Objekte mit dem Aufbewahren-bis-Datum bis zum 30. Januar 2020 an

```
public String createGovernanceRetentionJob(final AWSS3ControlClient awss3ControlClient)
 throws ParseException {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/governance-objects-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/governance-objects";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
    final Date jan30th = format.parse("30/01/2020");

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()
                            .withMode(S3ObjectLockRetentionMode.GOVERNANCE)
                            .withRetainUntilDate(jan30th)));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
```

```
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Put governance retention")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Example Umgehen Sie Aufbewahrungs-Governance über mehrere Objekte

Das folgende Beispiel baut auf dem vorherigen Beispiel zum Erstellen einer Vertrauensrichtlinie sowie zum Festlegen von S3-Batchoperations- und S3-Objektsperren-Konfigurations-Berechtigungen auf. Es zeigt, wie Sie die Aufbewahrungs-Governance über mehrere Objekte hinweg umgehen und erstellt einen Batchoperations-Auftrag, der den Manifest-Bucket verwendet und die Ergebnisse im Berichts-Bucket meldet.

```
public void allowBypassGovernance() {
    final String roleName = "bops-object-lock";

    final String bypassGovernancePermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:BypassGovernanceRetention\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket/*\"" +
            "            ]" +
            "        }" +
            "    ]" +
            "}";
```

```
    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withPolicyDocument(bypassGovernancePermissions)
            .withPolicyName("bypass-governance-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
public String createRemoveGovernanceRetentionJob(final AWSS3ControlClient
 awss3ControlClient) {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/governance-objects-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/bops-governance";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()));
```

```
    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Remove governance retention")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Verwendung rechtlicher Aufbewahrungsfristen für die S3-Objektsperre mithilfe von S3-Batch-Vorgänge deaktivieren

Das folgende Beispiel baut auf den vorherigen Beispielen zum Erstellen einer Vertrauensrichtlinie sowie zum Festlegen von S3-Batchoperations- und S3-Objektsperren-Konfigurations-Berechtigungen auf. Es zeigt, wie Sie die rechtliche Aufbewahrungsfrist für die Objektsperre für Objekte mithilfe von Batch-Vorgänge deaktivieren können.

Im Beispiel wird zuerst die Rolle aktualisiert, um `s3:PutObjectLegalHold`-Berechtigungen zu erteilen, ein Batchoperations-Auftrag erstellt, der die rechtliche Aufbewahrungsfrist von den im Manifest identifizierten Objekten deaktiviert (entfernt) und anschließend Berichte darüber erstellt.

Verwendung der AWS CLI

Example Aktualisiert die Rolle, um **`s3:PutObjectLegalHold`** Berechtigungen zu erteilen

```
export AWS_PROFILE='aws-user'

read -d '' legal_hold_permissions <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectLegalHold"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        }
    ]

EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name legal-hold-
permissions --policy-document "${legal_hold_permissions}"
```

Example Deaktivieren der gesetzlichen Aufbewahrungsfrist

Im folgenden Beispiel wird die rechtliche Aufbewahrungsfrist deaktiviert.

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectLegalHold": {
    "LegalHold": {
      "Status":"OFF"
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
```

```
    "ObjectArn": "arn:aws:s3:::ManifestBucket/legalhold-object-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucket",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/legalhold-objects-bops",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Turn off legal hold";
```

Verwenden des AWS-SDK für Java

Example Aktualisiert die Rolle, um **`s3:PutObjectLegalHold`** Berechtigungen zu erteilen

```
public void allowPutObjectLegalHold() {
    final String roleName = "bops-object-lock";

    final String legalHoldPermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:PutObjectLegalHold\"" +
            "            ]," +
            "            \"Resource\": [" +
```

```
            "                \"arn:aws:s3:::ManifestBucket/*\"" +
            "            ]" +
            "        }" +
            "    ]" +
            "}";

    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withPolicyDocument(legalHoldPermissions)
            .withPolicyName("legal-hold-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
```

Example Deaktivieren der gesetzlichen Aufbewahrungsfrist

Verwenden Sie das folgende Beispiel, wenn Sie die rechtliche Aufbewahrungsfrist deaktivieren möchten.

```
public String createLegalHoldOffJob(final AWSS3ControlClient awss3ControlClient) {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/legalhold-object-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/legalhold-objects-bops";
```

```
    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectLegalHold(new S3SetObjectLegalHoldOperation()
                    .withLegalHold(new S3ObjectLockLegalHold()
                            .withStatus(S3ObjectLockLegalHoldStatus.OFF)));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Turn off legal hold")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Tutorial zu S3-Batchvorgängen

Im folgenden Tutorial werden vollständige End-to-End-Verfahren für einige Batchvorgänge vorgestellt.

- Tutorial: Batch-Transcodierung von Videos mit S3- AWS Lambda Batchoperationen und AWS Elemental MediaConvert

Überwachen von Amazon S3

Überwachung ist wichtig, um die Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon S3 und Ihrer - AWS Lösungen aufrechtzuerhalten. Wir empfehlen, Überwachungsdaten von allen Teilen Ihrer - AWS Lösung zu sammeln, damit Sie einen Multipoint-Fehler leichter debuggen können, falls ein solcher auftritt. Bevor Sie mit der Überwachung von Amazon S3 beginnen, sollten Sie einen Überwachungsplan mit Antworten auf die folgenden Fragen erstellen:

- Was sind Ihre Ziele bei der Überwachung?
- Welche Ressourcen werden überwacht?
- Wie oft werden diese Ressourcen überwacht?
- Welche Überwachungstools werden verwendet?
- Wer soll die Überwachungsaufgaben ausführen?
- Wer soll benachrichtigt werden, wenn Fehler auftreten?

Weitere Informationen zur Protokollierung und Überwachung in Amazon S3 finden Sie in den folgenden Themen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Überwachungstools

AWS bietet verschiedene Tools, mit denen Sie Amazon S3 überwachen können. Sie können einige dieser Tools so konfigurieren, dass diese die Überwachung für Sie übernehmen, während bei anderen Tools ein manuelles Eingreifen nötig ist. Wir empfehlen, dass Sie die Überwachungsaufgaben möglichst automatisieren.

Automatisierte Überwachungstools

Sie können die folgenden automatisierten Tools zur Überwachung von Amazon S3 verwenden und auftretende Probleme melden:

- Amazon CloudWatch -Alarme – Überwachen Sie eine einzelne Metrik über einen von Ihnen angegebenen Zeitraum und führen Sie eine oder mehrere Aktionen aus, die auf dem Wert der Metrik im Verhältnis zu einem bestimmten Schwellenwert über eine Reihe von Zeiträumen basieren. Die Aktion ist eine Benachrichtigung, die an ein Amazon Simple Notification Service (Amazon SNS)-Thema oder eine Amazon EC2 Auto Scaling-Richtlinie gesendet wird. - CloudWatch Alarme rufen keine Aktionen auf, nur weil sie sich in einem bestimmten Status befinden. Der Status muss sich geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.
- AWS CloudTrail Protokollüberwachung – Teilen Sie Protokolldateien zwischen Konten, überwachen Sie CloudTrail Protokolldateien in Echtzeit, indem Sie sie an - CloudWatch Protokolle senden, schreiben Sie Anwendungen zur Protokollverarbeitung in Java und überprüfen Sie, ob sich Ihre Protokolldateien nach der Bereitstellung durch nicht geändert haben CloudTrail. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail.

Manuelle Überwachungstools

Ein weiterer wichtiger Bestandteil der Überwachung von Amazon S3 ist die manuelle Überwachung derjenigen Elemente, die die CloudWatch Alarme nicht abdecken. Amazon S3 CloudWatch Trusted Advisor, und andere AWS Management Console Dashboards bieten einen at-a-glance Überblick über den Zustand Ihrer AWS Umgebung. Möglicherweise wollen Sie die Server-Zugriffsprotokollierung aktivieren, die Zugriffsanforderungen für den Bucket verfolgt. Jeder Zugriffsprotokolldatensatz enthält Details über eine Zugriffsanforderung, z. B. Auftraggeber, Bucket-Name, Anforderungszeit, Anforderungsaktion, Antwortstatus und Fehlercode, falls vorhanden. Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

- Auf dem Amazon-S3-Dashboard wird Folgendes angezeigt:
  - Ihre Buckets und die Eigenschaften und Objekte, die sie enthalten.
- Auf der - CloudWatch Startseite wird Folgendes angezeigt:
  - Aktuelle Alarme und Status
  - Diagramme mit Alarmen und Ressourcen
  - Servicestatus

  Darüber hinaus können Sie mit Folgendes CloudWatch tun:
  - Erstellen von benutzerdefinierten Dashboards zur Überwachung des gewünschten Services.
  - Aufzeichnen von Metrikdaten, um Probleme zu beheben und Trends zu erkennen.
  - Suchen und durchsuchen Sie alle Ihre AWS Ressourcenmetriken.
  - Erstellen und Bearbeiten von Alarmen, um über Probleme benachrichtigt zu werden.
- AWS Trusted Advisor kann Ihnen helfen, Ihre - AWS Ressourcen zu überwachen, um Leistung, Zuverlässigkeit, Sicherheit und Kosteneffizienz zu verbessern. Vier Trusted Advisor -Prüfungen stehen allen Benutzern zur Verfügung; mehr als 50 Überprüfungen stehen Benutzern mit einem Business- oder Enterprise-Supportplan zur Verfügung. Weitere Informationen finden Sie unter AWS Trusted Advisor.

  Trusted Advisor führt die folgenden Prüfungen im Zusammenhang mit Amazon S3 durch:
  - Überprüfungen der Protokollierungskonfiguration von Amazon-S3-Buckets.
  - Sicherheitsprüfungen für Amazon-S3-Buckets mit offenen Zugriffsberechtigungen.
  - Fehlertoleranzprüfungen für Amazon-S3-Buckets, für die kein Versioning aktiviert oder deren Versioning ausgesetzt ist.

Protokollierungsoptionen für Amazon S3

Sie können die Aktionen aufzeichnen, die von Benutzern, Rollen oder AWS-Services Amazon S3-Ressourcen durchgeführt werden, und Protokolldatensätze zu Prüfungs- und Compliance-Zwecken verwalten. Dazu können Sie die Serverzugriffsprotokollierung, die AWS CloudTrail - Protokollierung oder eine Kombination aus beiden verwenden. Wir empfehlen Ihnen, CloudTrail für die Protokollierung von Aktionen auf Bucket- und Objektebene für Ihre Amazon S3-Ressourcen zu verwenden. Weitere Informationen zu jeder Option finden Sie in folgenden Abschnitten:

- Protokollieren von Anfragen mit Server-Zugriffsprotokollierung

- Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail

In der folgenden Tabelle sind die wichtigsten Eigenschaften von CloudTrail Protokollen und Amazon S3-Serverzugriffsprotokollen aufgeführt. Um sicherzustellen, dass Ihre Sicherheitsanforderungen CloudTrail erfüllt, lesen Sie die Tabelle und die Hinweise.

Protokolleigenschaften	AWS CloudTrail	Amazon-S3-Serverprotokolle
Kann an andere Systeme weitergeleitet werden (Amazon CloudWatch Logs, Amazon CloudWatch Events)	Ja	No
Protokolle an mehreren Zielen bereitstellen (Beispiel: dieselben Protokolle an zwei verschiedene Buckets senden)	Ja	No
Protokolle für eine Teilmenge von Objekten aktivieren (Präfix)	Ja	No
Kontoübergreifende Protokoll bereitstellung (Ziel- und Quell-Bucket in Besitz von verschied enen Konten)	Ja	No
Integritätsprüfung der Protokoll datei mit digitaler Signatur oder Hashing	Ja	No
Standardeinstellung oder Auswahl der Verschlüsselung für Protokolldateien	Ja	No
Objektvorgänge (mit Amazon-S3-APIs)	Ja	Ja

Protokolleigenschaften	AWS CloudTrail	Amazon-S3-Serverprotokolle
Bucket-Vorgänge (mit Amazon-S3-APIs)	Ja	Ja
Durchsuchbare UI für Protokolle	Ja	No
Felder für Objektsperrparameter, Amazon S3 Select-Eigenschaften für Protokolldatensätze	Ja	No
Felder für `Object Size`, `Total Time`, `Turn-Around Time` und `HTTP Referer` für Protokolldatensätze		Ja
Lebenszyklusübertragungen, Ablaufaktionen, Wiederherstellungen		Ja
Protokollieren von Schlüsseln in einer Batch-Delete-Operation		Ja
Authentifizierungsfehler[1]		Ja
Konten, an die Protokolle geliefert werden	Bucket-Eigentümer[2] und Auftraggeber	Nur Bucket-Eigentümer
Performance and Cost	AWS CloudTrail	Amazon S3 Server Logs
Preis	Verwaltungsereignisse (erste Bereitstellung) sind kostenlos. Für Datenereignisse fällt zusätzlich zur Speicherung der Protokolle eine Gebühr an	Keine Zusatzkosten neben der Speicherung der Protokolle

Protokolleigenschaften	AWS CloudTrail	Amazon-S3-Serverprotokolle
Geschwindigkeit der Protokoll bereitstellung	Datenereignisse alle 5 Minuten; Verwaltungsereignisse alle 15 Minuten	Innerhalb weniger Stunden
Protokollformat	JSON	Protokolldatei mit durch Leerzeichen getrennten, durch neue Zeilen getrennten Datensätzen

Hinweise

1. CloudTrail liefert keine Protokolle für Anforderungen, bei denen die Authentifizierung fehlschlägt (in denen die bereitgestellten Anmeldeinformationen nicht gültig sind). Es enthält jedoch Protokolle für Anforderungen, deren Authentifizierung fehlschlägt (`AccessDenied`), und Anforderungen, die von anonymen Benutzern gestellt werden.
2. Der S3-Bucket-Eigentümer erhält CloudTrail Protokolle, wenn das Konto keinen vollständigen Zugriff auf das Objekt in der Anforderung hat. Weitere Informationen finden Sie unter Aktionen auf Objektebene in kontoübergreifenden Szenarien.
3. S3 unterstützt nicht die Bereitstellung von CloudTrail Protokollen oder Serverzugriffsprotokollen an den Anforderer oder den Bucket-Eigentümer für VPC-Endpunktanforderungen, wenn die VPC-Endpunktrichtlinie diese verweigert.

Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail

Amazon S3 ist in integriert, einem Service AWS CloudTrail, der die Aktionen eines Benutzers, einer Rolle oder eines - AWS Services in Amazon S3 aufzeichnet. CloudTrail erfasst eine Teilmenge der API-Aufrufe für Amazon S3 als Ereignisse, einschließlich Aufrufen von der Amazon S3-Konsole und Codeaufrufen an Amazon S3-API-Operationen.

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3-Bucket aktivieren, einschließlich Ereignissen für Amazon S3. Wenn Sie keinen Trail konfigurieren, können Sie trotzdem die neuesten Ereignisse in der CloudTrail Konsole unter Ereignisverlauf anzeigen. Anhand der von CloudTrailgesammelten Informationen

können Sie die an Amazon S3 gestellte Anfrage, die IP-Adresse, von der die Anfrage gestellt wurde, den Initiator der Anfrage, den Zeitpunkt der Anfrage und zusätzliche Details bestimmen.

Weitere Informationen zu CloudTrail, einschließlich Konfiguration und Aktivierung, finden Sie im AWS CloudTrail -Benutzerhandbuch.

Verwenden von CloudTrail Protokollen mit Amazon S3-Serverzugriffsprotokollen und CloudWatch -Protokollen

AWS CloudTrail -Protokolle bieten eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem AWS -Service in Amazon S3 durchgeführten Aktionen, während Amazon S3-Serverzugriffsprotokolle detaillierte Aufzeichnungen über die Anforderungen bereitstellen, die an einen S3-Bucket gestellt werden. Weitere Informationen zur Funktionsweise der unterschiedlichen Protokolle und ihren Eigenschaften, ihrer Leistung und ihrer Kosten finden Sie unter the section called “Protokollierungsoptionen”.

Sie können - AWS CloudTrail Protokolle zusammen mit Serverzugriffsprotokollen für Amazon S3 verwenden. - CloudTrail Protokolle bieten Ihnen eine detaillierte API-Nachverfolgung für Amazon-Amazon S3-Operationen auf Bucket- und Objektebene. Serverzugriffsprotokolle für Amazon S3 bieten Ihnen Einblicke in Operationen auf Objektebene für Ihre Daten in Amazon S3. Weitere Informationen zu Server-Zugriffsprotokollen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

Sie können - CloudTrail Protokolle auch zusammen mit Amazon CloudWatch für Amazon S3 verwenden. CloudTrail Integration mit - CloudWatch Protokollen liefert von erfasste API-Aktivitäten auf S3-Bucket-Ebene CloudTrail an einen CloudWatch Protokollstream in der von Ihnen angegebenen CloudWatch Protokollgruppe. Sie können CloudWatch Alarme für die Überwachung bestimmter API-Aktivitäten erstellen und E-Mail-Benachrichtigungen erhalten, wenn die spezifische API-Aktivität stattfindet. Weitere Informationen zu CloudWatch Alarmen für die Überwachung bestimmter API-Aktivitäten finden Sie im AWS CloudTrail -Benutzerhandbuch. Weitere Informationen zur Verwendung von CloudWatch mit Amazon S3 finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

> **Note**
>
> S3 unterstützt keine Übermittlung von CloudTrail Protokollen an den Anforderer oder den Bucket-Eigentümer für VPC-Endpunktanforderungen, wenn die VPC-Endpunktrichtlinie sie ablehnt.

CloudTrail -Nachverfolgung mit Amazon S3-SOAP-API-Aufrufen

CloudTrail verfolgt Amazon S3-SOAP-API-Aufrufe. Die Amazon S3 SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Weitere Informationen über die Amazon S3 SOAP-Unterstützung erhalten Sie unter Anhang A: Verwenden der SOAP-API.

> **Important**
>
> Neuere Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen Ihnen, entweder die REST-API oder die - AWS SDKs zu verwenden.

Amazon S3-SOAP-Aktionen, die durch die CloudTrail Protokollierung verfolgt werden

SOAP API-Name	Im CloudTrail Protokoll verwendeter API-Ereignisname
ListAllMyBuckets	`ListBuckets`
CreateBucket	`CreateBucket`
DeleteBucket	`DeleteBucket`
GetBucketAccessControlPolicy	`GetBucketAcl`
SetBucketAccessControlPolicy	`PutBucketAcl`
GetBucketLoggingStatus	`GetBucketLogging`
SetBucketLoggingStatus	`PutBucketLogging`

Weitere Informationen zu CloudTrail und Amazon S3 finden Sie in den folgenden Themen:

Themen

- Amazon S3 CloudTrail -Ereignisse
- CloudTrail -Protokolldateieinträge für Amazon S3 und S3 on Outposts
- Aktivieren der CloudTrail Ereignisprotokollierung für S3-Buckets und -Objekte
- Identifizieren von Amazon S3-Anforderungen mit CloudTrail

Amazon S3 CloudTrail -Ereignisse

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

CloudTrail wird beim Erstellen des Kontos AWS-Konto auf Ihrem aktiviert. Wenn die unterstützte Ereignisaktivität in Amazon S3 auftritt, wird diese Aktivität in einem - CloudTrail Ereignis zusammen mit anderen - AWS Serviceereignissen im Ereignisverlauf aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem anzeigen, suchen und herunterladen AWS-Konto. Weitere Informationen finden Sie unter Anzeigen von Ereignissen mit dem CloudTrail Ereignisverlauf.

Erstellen Sie für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, einschließlich Ereignissen für Amazon S3, einen Trail. Ein Trail ermöglicht CloudTrail die Bereitstellung von Protokolldateien an einen Amazon S3-Bucket. Wenn Sie ein Trail in der Konsole anlegen, gilt dieser für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS -Partition und stellt die Protokolldateien in dem von Ihnen angegebenen Amazon S3-Bucket bereit. Darüber hinaus können Sie andere - AWS Services konfigurieren, um die in den CloudTrail Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu agieren. Weitere Informationen finden Sie hier:

- Erstellen eines Trails für Ihr AWS-Konto
- AWS-Services -Integrationen mit - CloudTrail Protokollen
- Konfigurieren von Amazon SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen
- Empfangen von CloudTrail Protokolldateien von mehreren Konten

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit Anmeldeinformationen des Root-Benutzers oder des IAM-Benutzers gestellt wurde
- Ob die Anfrage mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen föderierten Benutzer ausgeführt wurde
- Ob die Anforderung von einem anderen gestellt wurde AWS-Service

Weitere Informationen hierzu finden Sie unter dem CloudTrail-Element userIdentity.

Sie können Ihre Protokolldateien beliebig lange im Bucket speichern. Sie können aber auch Amazon S3-Lebenszyklusregeln aufstellen, anhand derer die Protokolldateien automatisch archiviert oder gelöscht werden. Standardmäßig werden die Protokolldateien mit serverseitiger Amazon S3-Verschlüsselung (SSE) verschlüsselt.

Wie Anforderungen an Amazon S3 CloudTrail erfasst

Standardmäßig CloudTrail protokolliert API-Aufrufe auf S3-Bucket-Ebene, die in den letzten 90 Tagen getätigt wurden, aber keine Anforderungen an Objekte. Aufrufe auf Bucket-Ebene sind Ereignisse wie `CreateBucket`, `DeleteBucket`, `PutBucketLifecycle`, `PutBucketPolicy` usw. Sie können Ereignisse auf Bucket-Ebene in der - CloudTrail Konsole anzeigen. Sie können dort jedoch keine Datenereignisse (Aufrufe auf Amazon S3-Objektebene) anzeigen – Sie müssen Protokolle für sie analysieren oder abfragen CloudTrail.

Amazon S3-Aktionen auf Kontoebene, die durch die CloudTrail Protokollierung verfolgt werden

CloudTrail protokolliert Aktionen auf Kontoebene. Amazon S3-Datensätze werden zusammen mit anderen AWS-Service Datensätzen in einer Protokolldatei geschrieben. CloudTrail bestimmt, wann eine neue Datei basierend auf einem Zeitraum und einer Dateigröße erstellt und in eine neue Datei geschrieben werden soll.

In den Tabellen in diesem Abschnitt sind die AmazonAmazon S3-Aktionen auf Kontoebene aufgeführt, die für die Protokollierung durch unterstützt werden CloudTrail.

API-Aktionen auf Kontoebene von Amazon S3, die durch die CloudTrail Protokollierung verfolgt werden, werden als die folgenden Ereignisnamen angezeigt. Die CloudTrail Ereignisnamen

unterscheiden sich vom Namen der API-Aktion. Zum Beispiel DeletePublicAccessBlock ist DeleteAccountPublicAccessBlock.

- DeleteAccountPublicAccessBlock
- GetAccountPublicAccessBlock
- PutAccountPublicAccessBlock

Amazon S3-Aktionen auf Bucket-Ebene, die durch die CloudTrail Protokollierung verfolgt werden

Standardmäßig CloudTrail protokolliert Aktionen auf Bucket-Ebene für Allzweck-Buckets. Amazon S3-Datensätze werden zusammen mit anderen AWS Service-Datensätzen in einer Protokolldatei geschrieben. CloudTrail bestimmt anhand eines Zeitraums und einer Dateigröße, wann eine neue Datei erstellt und in diese geschrieben werden soll.

In diesem Abschnitt werden die Aktionen auf Amazon S3-Bucket-Ebene aufgeführt, die für die Protokollierung von unterstützt werden CloudTrail.

API-Aktionen auf Amazon S3-Bucket-Ebene, die durch die - CloudTrail Protokollierung verfolgt werden, werden als die folgenden Ereignisnamen angezeigt. In einigen Fällen unterscheidet sich der CloudTrail Ereignisname vom Namen der API-Aktion. Zum Beispie, `PutBucketLifecycleConfiguration` ist `PutBucketLifecycle`.

- CreateBucket
- DeleteBucket
- DeleteBucketAnalyticsConfiguration
- DeleteBucketCors
- DeleteBucketEncryption
- DeleteBucketIntelligentTieringConfiguration
- DeleteBucketInventoryConfiguration
- DeleteBucketLifecycle
- DeleteBucketMetricsConfiguration
- DeleteBucketOwnershipControls
- DeleteBucketPolicy
- DeleteBucketPublicAccessBlock

- DeleteBucketReplication
- DeleteBucketTagging
- GetAccelerateConfiguration
- GetBucketAcl
- GetBucketAnalyticsConfiguration
- GetBucketCors
- GetBucketEncryption
- GetBucketIntelligentTieringConfiguration
- GetBucketInventoryConfiguration
- GetBucketLifecycle
- GetBucketLocation
- GetBucketLogging
- GetBucketMetricsConfiguration
- GetBucketNotification
- GetBucketObjectLockConfiguration
- GetBucketOwnershipControls
- GetBucketPolicy
- GetBucketPolicyStatus
- GetBucketPublicAccessBlock
- GetBucketReplication
- GetBucketRequestPayment
- GetBucketTagging
- GetBucketVersioning
- GetBucketWebsite
- HeadBucket
- ListBuckets
- PutAccelerateConfiguration
- PutBucketAcl
- PutBucketAnalyticsConfiguration
- PutBucketCors

- PutBucketEncryption
- PutBucketIntelligentTieringConfiguration
- PutBucketInventoryConfiguration
- PutBucketLifecycle
- PutBucketLogging
- PutBucketMetricsConfiguration
- PutBucketNotification
- PutBucketObjectLockConfiguration
- PutBucketOwnershipControls
- PutBucketPolicy
- PutBucketPublicAccessBlock
- PutBucketReplication
- PutBucketRequestPayment
- PutBucketTagging
- PutBucketVersioning
- PutBucketWebsite

Zusätzlich zu diesen API-Operationen können Sie auch die Objektebenenaktion OPTIONS-Objekt verwenden. Diese Aktion wird in der CloudTrail Protokollierung wie eine Aktion auf Bucket-Ebene behandelt, da die Aktion die CORS-Konfiguration eines Buckets überprüft.

S3-Express-One-Zone-Aktionen auf Bucket-Ebene (regionaler API-Endpunkt), die durch CloudTrail Protokollierung verfolgt werden

Standardmäßig CloudTrail protokolliert Aktionen auf Bucket-Ebene für Verzeichnis-Buckets als Verwaltungsereignisse. Die `eventsource` für CloudTrail Verwaltungsereignisse für S3 Express One Zone ist `s3express.amazonaws.com`.

> **Note**
>
> Für S3 Express One Zone wird CloudTrail die Protokollierung von API-Operationen für zonale Endpunkte (Objektebene oder Datenebene) (z. B. `PutObject` oder `GetObject`) nicht unterstützt.

Die folgenden API-Operationen für regionale Endpunkte werden in protokolliert CloudTrail.

- CreateBucket
- DeleteBucket
- DeleteBucketPolicy
- GetBucketPolicy
- PutBucketPolicy
- ListDirectoryBuckets

Weitere Informationen finden Sie unter Bewährte Methoden für die Sicherheit in S3 Express One Zone.

Amazon S3-Aktionen auf Objektebene, die durch die AWS CloudTrail Protokollierung verfolgt werden

Sie können auch CloudTrail Protokolle für Amazon S3-Aktionen auf Objektebene abrufen. Aktivieren Sie dazu Datenereignisse für Ihren S3-Bucket oder alle Buckets in Ihrem Konto. Wenn eine Aktion auf Objektebene in Ihrem Konto stattfindet, CloudTrail wertet Ihre Trail-Einstellungen aus. Falls das Ereignis mit dem von Ihnen angegebenen Objekt in einem Pfad übereinstimmt, wird das Ereignis protokolliert. Weitere Informationen finden Sie unter Aktivieren der CloudTrail Ereignisprotokollierung für S3-Buckets und -Objekte und Protokollieren von Datenereignissen für Trails im AWS CloudTrail - Benutzerhandbuch.

> **Note**
>
> S3 unterstützt nicht die Übermittlung von CloudTrail Protokollen an den Anforderer oder den Bucket-Eigentümer für VPC-Endpunktanforderungen, wenn die VPC-Endpunktrichtlinie sie ablehnt.

API-Aktionen auf Objektebene von Amazon S3, die durch die CloudTrail Protokollierung verfolgt werden, werden als die folgenden Ereignisnamen angezeigt. In einigen Fällen unterscheidet sich der CloudTrail Ereignisname vom Namen der API-Aktion.

- AbortMultipartUpload
- CompleteMultipartUpload

- CopyObject
- CreateMultipartUpload
- DeleteObject
- DeleteObjectTagging
- DeleteObjects
- GetObject
- GetObjectAcl
- GetObjectAttributes
- GetObjectLockLegalHold
- GetObjectLockRetention
- GetObjectTagging
- GetObjectTorrent
- HeadObject
- ListMultipartUploads
- ListObjectVersions
- ListObjects
- ListParts
- PutObject
- PutObjectAcl
- PutObjectLockLegalHold
- PutObjectLockRetention
- PutObjectTagging
- RestoreObject
- SelectObjectContent
- UploadPart
- UploadPartCopy

Aktionen auf Objektebene in kontoübergreifenden Szenarien

Im Folgenden finden Sie spezielle Anwendungsfälle, bei denen API-Aufrufe auf Objektebene in kontoübergreifenden Szenarien durchgeführt werden und wie CloudTrail Protokolle gemeldet werden.

CloudTrail liefert Protokolle an den Anforderer (das Konto, das den API-Aufruf getätigt hat), außer in einigen Fällen, in denen Protokolleinträge redigiert oder weggelassen werden. Bei der Einrichtung von kontoübergreifendem Zugriff sehen Sie sich die Beispiele in diesem Abschnitt an.

> **Note**
>
> In den Beispielen wird davon ausgegangen, dass CloudTrail Protokolle ordnungsgemäß konfiguriert sind.

Beispiel 1: CloudTrail liefert Protokolle an den Bucket-Eigentümer

CloudTrail liefert Protokolle an den Bucket-Eigentümer, auch wenn der Bucket-Eigentümer keine Berechtigungen für dieselbe Objekt-API-Operation hat. Sehen Sie sich das folgende kontenübergreifende Szenario vor:

- Konto A ist Eigentümer des Buckets.
- Konto B (Anforderer) versucht, auf ein Objekt in diesem Bucket zuzugreifen.
- Konto C besitzt das Objekt. Konto C kann dasselbe Konto wie Konto A sein oder auch nicht.

> **Note**
>
> CloudTrail liefert API-Protokolle auf Objektebene immer an den Anforderer (Konto B). Darüber hinaus liefert dieselben Protokolle CloudTrail auch dann an den Bucket-Eigentümer (Konto A), wenn der Bucket-Eigentümer nicht Eigentümer des Objekts (Konto C) ist oder über Berechtigungen für dieselben API-Operationen für dieses Objekt verfügt.

Beispiel 2: CloudTrail gibt keine E-Mail-Adressen weiter, die beim Festlegen von Objekt-ACLs verwendet werden

Sehen Sie sich das folgende kontenübergreifende Szenario vor:

- Konto A ist Eigentümer des Buckets.
- Konto B (Anforderer) sendet eine Anforderung, um eine ACL-Erteilung für ein Objekt unter Verwendung einer E-Mail-Adresse einzurichten. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht.

Der Anforderer erhält die Protokolle zusammen mit der E-Mail-Information. Der Bucket-Eigentümer erhält jedoch, wenn er berechtigt ist, Protokolle zu empfangen, wie in Beispiel 1, das CloudTrail Protokoll, das das Ereignis meldet. Der Bucket-Eigentümer erhält jedoch keine Informationen über die ACL-Konfiguration, insbesondere die E-Mail des Empfängers und die Erteilung. Die einzige Information, die das Protokoll dem Bucket-Eigentümer mitteilt, ist, dass Konto B einen ACL-API-Aufruf vorgenommen hat.

CloudTrail -Protokolldateieinträge für Amazon S3 und S3 on Outposts

> **Important**
>
> Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der automatische Verschlüsselungsstatus für die Standardverschlüsselungskonfiguration des S3-Buckets und für neue Objekt-Uploads ist in - AWS CloudTrail Protokollen, S3 Inventory, S3 Storage Lens, der Amazon S3-Konsole und als zusätzlicher Amazon S3-API-Antwort-Header in der AWS Command Line Interface und AWS SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Ein Trail ist eine Konfiguration, die die Bereitstellung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar. Es enthält unter anderem Informationen über die angeforderte Aktion, etwaige Anforderungsparameter und das Datum und die Uhrzeit der Aktion. CloudTrail -Protokolldateien sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

Weitere Informationen finden Sie in den folgenden Beispielen.

Themen

- Beispiel: CloudTrail Protokolldateieintrag für Amazon S3
- Beispiel: Amazon S3 in Outposts-Logdateieinträgen

Beispiel: CloudTrail Protokolldateieintrag für Amazon S3

Das folgende Beispiel zeigt einen - CloudTrail Protokolleintrag, der die GetBucketVersioning Aktionen GETPutBucketAcl, und für Service demonstriert.


```
{
    "Records": [
    {
        "eventVersion": "1.03",
        "userIdentity": {
            "type": "IAMUser",
            "principalId": "111122223333",
            "arn": "arn:aws:iam::111122223333:user/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "userName": "myUserName"
        },
        "eventTime": "2019-02-01T03:18:19Z",
        "eventSource": "s3.amazonaws.com",
        "eventName": "ListBuckets",
        "awsRegion": "us-west-2",
        "sourceIPAddress": "127.0.0.1",
        "userAgent": "[]",
        "requestParameters": {
            "host": [
                "s3.us-west-2.amazonaws.com"
            ]
        },
        "responseElements": null,
        "additionalEventData": {
            "SignatureVersion": "SigV2",
            "AuthenticationMethod": "QueryString",
            "aclRequired": "Yes"
    },
        "requestID": "47B8E8D397DCE7A6",
        "eventID": "cdc4b7ed-e171-4cef-975a-ad829d4123e8",
        "eventType": "AwsApiCall",
        "recipientAccountId": "444455556666",
        "tlsDetails": {
            "tlsVersion": "TLSv1.2",
            "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
            "clientProvidedHostHeader": "s3.amazonaws.com"
    }
```

```
    },
    {
       "eventVersion": "1.03",
       "userIdentity": {
            "type": "IAMUser",
            "principalId": "111122223333",
            "arn": "arn:aws:iam::111122223333:user/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "userName": "myUserName"
        },
      "eventTime": "2019-02-01T03:22:33Z",
      "eventSource": "s3.amazonaws.com",
      "eventName": "PutBucketAcl",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "",
      "userAgent": "[]",
      "requestParameters": {
          "bucketName": "",
          "AccessControlPolicy": {
              "AccessControlList": {
                  "Grant": {
                      "Grantee": {
                          "xsi:type": "CanonicalUser",
                          "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
                          "ID":
"d25639fbe9c19cd30a4c0f43fbf00e2d3f96400a9aa8dabfbbebe1906Example"
                       },
                      "Permission": "FULL_CONTROL"
                   }
              },
              "xmlns": "http://s3.amazonaws.com/doc/2006-03-01/",
              "Owner": {
                  "ID":
"d25639fbe9c19cd30a4c0f43fbf00e2d3f96400a9aa8dabfbbebe1906Example"
              }
          },
          "host": [
              "s3.us-west-2.amazonaws.com"
          ],
          "acl": [
              ""
          ]
      },
```

```
      "responseElements": null,
      "additionalEventData": {
          "SignatureVersion": "SigV4",
          "CipherSuite": "ECDHE-RSA-AES128-SHA",
          "AuthenticationMethod": "AuthHeader"
      },
      "requestID": "BD8798EACDD16751",
      "eventID": "607b9532-1423-41c7-b048-ec2641693c47",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333",
      "tlsDetails": {
            "tlsVersion": "TLSv1.2",
            "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
            "clientProvidedHostHeader": "s3.amazonaws.com"
    }
    },
    {
      "eventVersion": "1.03",
      "userIdentity": {
          "type": "IAMUser",
          "principalId": "111122223333",
          "arn": "arn:aws:iam::111122223333:user/myUserName",
          "accountId": "111122223333",
          "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
          "userName": "myUserName"
        },
      "eventTime": "2019-02-01T03:26:37Z",
      "eventSource": "s3.amazonaws.com",
      "eventName": "GetBucketVersioning",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "",
      "userAgent": "[]",
      "requestParameters": {
          "host": [
              "s3.us-west-2.amazonaws.com"
          ],
          "bucketName": "DOC-EXAMPLE-BUCKET1",
          "versioning": [
              ""
          ]
      },
      "responseElements": null,
      "additionalEventData": {
          "SignatureVersion": "SigV4",
```

```
          "CipherSuite": "ECDHE-RSA-AES128-SHA",
          "AuthenticationMethod": "AuthHeader"
    },
      "requestID": "07D681279BD94AED",
      "eventID": "f2b287f3-0df1-4961-a2f4-c4bdfed47657",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333",
      "tlsDetails": {
            "tlsVersion": "TLSv1.2",
            "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
            "clientProvidedHostHeader": "s3.amazonaws.com"
    }
    }
  ]
}
```

Beispiel: Amazon S3 in Outposts-Logdateieinträgen

Verwaltungsereignisse für Amazon S3 on Outposts sind über verfügbar AWS CloudTrail. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail. Darüber hinaus können Sie optional die Protokollierung für Datenereignisse aktivieren in AWS CloudTrail.

Ein Trail ist eine Konfiguration, durch die Ereignisse als Protokolldateien an den von Ihnen angegebenen S3-Bucket in einer Region übermittelt werden. CloudTrail -Protokolle für Ihre Outposts-Buckets enthalten ein neues Feld, `edgeDeviceDetails`, das den Outpost identifiziert, in dem sich der angegebene Bucket befindet.

Zu den zusätzlichen Protokollfeldern gehören die angeforderte Aktion, das Datum und die Uhrzeit der Aktion, und die Dateien request parameters. CloudTrail log sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

Das folgende Beispiel zeigt einen - CloudTrail Protokolleintrag, der eine PutObject Aktion auf demonstriert`s3-outposts`.

```
{
      "eventVersion": "1.08",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "111122223333",
        "arn": "arn:aws:iam::111122223333:user/yourUserName",
        "accountId": "222222222222",
```

```
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "yourUserName"
      },
      "eventTime": "2020-11-30T15:44:33Z",
      "eventSource": "s3-outposts.amazonaws.com",
      "eventName": "PutObject",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "26.29.66.20",
      "userAgent": "aws-cli/1.18.39 Python/3.4.10 Darwin/18.7.0 botocore/1.15.39",
      "requestParameters": {
        "expires": "Wed, 21 Oct 2020 07:28:00 GMT",
        "Content-Language": "english",
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "ObjectCannedACL": "BucketOwnerFullControl",
        "x-amz-server-side-encryption": "Aes256",
        "Content-Encoding": "gzip",
        "Content-Length": "10",
        "Cache-Control": "no-cache",
        "Content-Type": "text/html; charset=UTF-8",
        "Content-Disposition": "attachment",
        "Content-MD5": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY",
        "x-amz-storage-class": "Outposts",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "bucketName": "DOC-EXAMPLE-BUCKET1",
        "Key": "path/upload.sh"
      },
      "responseElements": {
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "x-amz-server-side-encryption": "Aes256",
        "x-amz-version-id": "001",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "ETag": "d41d8cd98f00b204e9800998ecf8427f"
      },
      "additionalEventData": {
        "CipherSuite": "ECDHE-RSA-AES128-SHA",
        "bytesTransferredIn": 10,
        "x-amz-id-2": "29xXQBV20
+xOHKItvzY1suLv1i6A52E0zOX159fpfsItYd58JhXwKxXAXI4IQkp6",
        "SignatureVersion": "SigV4",
        "bytesTransferredOut": 20,
        "AuthenticationMethod": "AuthHeader"
      },
```

```
      "requestID": "8E96D972160306FA",
      "eventID": "ee3b4e0c-ab12-459b-9998-0a5a6f2e4015",
      "readOnly": false,
      "resources": [
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Object",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/path/upload.sh"
        },
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Bucket",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/"
        }
      ],
      "eventType": "AwsApiCall",
      "managementEvent": false,
      "recipientAccountId": "444455556666",
      "sharedEventID": "02759a4c-c040-4758-b84b-7cbaaf17747a",
      "edgeDeviceDetails": {
        "type": "outposts",
        "deviceId": "op-01ac5d28a6a232904"
      },
      "eventCategory": "Data"
    }
```

Aktivieren der CloudTrail Ereignisprotokollierung für S3-Buckets und -Objekte

Sie können CloudTrail Datenereignisse verwenden, um Informationen zu Anforderungen auf Bucket- und Objektebene in Amazon S3 abzurufen. Um CloudTrail Datenereignisse für alle Ihre Buckets oder für eine Liste bestimmter Buckets zu aktivieren, müssen Sie manuell einen Trail in erstellen CloudTrail.

> **Note**
>
> - Die Standardeinstellung für CloudTrail besteht darin, nur Verwaltungsereignisse zu finden. Prüfen Sie, ob Datenereignisse für das Konto aktiviert wurden.

- Ein S3-Bucket mit hoher Workload kann in kurzer Zeit Tausende Protokolle generieren. Beachten Sie, wie lange Sie CloudTrail Datenereignisse für einen ausgelasteten Bucket aktivieren möchten.

CloudTrail speichert Amazon S3-Datenereignisprotokolle in einem S3-Bucket Ihrer Wahl. Erwägen Sie, einen Bucket in einem separaten zu verwenden, AWS-Konto um Ereignisse aus mehreren Buckets, die Sie möglicherweise besitzen, besser an einem zentralen Ort zu organisieren und so die Abfrage und Analyse zu vereinfachen. AWS Organizations hilft Ihnen, ein zu erstellen AWS-Konto , das mit dem Konto verknüpft ist, dem der Bucket gehört, den Sie überwachen. Weitere Informationen finden Sie unter Was ist AWS Organizations? im AWS Organizations -Benutzerhandbuch.

Wenn Sie einen Trail in erstellen CloudTrail, können Sie im Abschnitt Datenereignisse das Kontrollkästchen Alle S3-Buckets in Ihrem Konto auswählen aktivieren, um alle Ereignisse auf Objektebene zu protokollieren.

Note

- Eine bewährte Methode besteht darin, eine Lebenszykluskonfiguration für Ihren Datenereignis-Bucket in AWS CloudTrail zu erstellen. Konfigurieren Sie die Lebenszykluskonfiguration zum regelmäßigen Entfernen von Protokolldateien nach dem Zeitraum, der Ihres Erachtens für die Überprüfung erforderlich ist. Dadurch wird die Menge der Daten reduziert, die Athena in einer Abfrage analysiert. Weitere Informationen finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.
- Weitere Informationen zum Format der Protokolle finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail.
- Beispiele für das Abfragen von CloudTrail Protokollen finden Sie im AWS -Big-Data-Blogbeitrag Analyze Security, Compliance, and Operational Activity Using AWS CloudTrail and Amazon Athena .

Aktivieren der Protokollierung für Objekte in einem Bucket mit der Konsole

Sie können die Amazon S3-Konsole verwenden, um einen - AWS CloudTrail Trail zum Protokollieren von Datenereignissen für Objekte in einem S3-Bucket zu konfigurieren. CloudTrail unterstützt die Protokollierung von Amazon S3-API-Operationen auf Objektebene wie `GetObject`, `DeleteObject`und `PutObject`. Diese Ereignisse werden als Datenereignisse bezeichnet.

Standardmäßig protokollieren CloudTrail Trails keine Datenereignisse, aber Sie können Trails so konfigurieren, dass Datenereignisse für von Ihnen angegebene S3-Buckets oder Datenereignisse für alle Amazon S3-Buckets in Ihrem protokolliert werden AWS-Konto. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail.

CloudTrail füllt keine Datenereignisse im CloudTrail Ereignisverlauf aus. Darüber hinaus werden nicht alle Aktionen auf Bucket-Ebene im CloudTrail Ereignisverlauf ausgefüllt. Weitere Informationen zu den Amazon S3-API-Aktionen auf Bucket-Ebene, die durch die CloudTrail Protokollierung verfolgt werden, finden Sie unter Amazon S3-Aktionen auf Bucket-Ebene, die durch die CloudTrail Protokollierung verfolgt werden. Weitere Informationen zum Abfragen von CloudTrail Protokollen finden Sie im AWS Knowledge-Center-Artikel über die Verwendung von Amazon- CloudWatch Logs-Filtermustern und Amazon Athena zum Abfragen von CloudTrail Protokollen.

Um einen Trail zum Protokollieren von Datenereignissen für einen S3-Bucket zu konfigurieren, können Sie entweder die AWS CloudTrail -Konsole oder die Amazon-S3-Konsole verwenden. Wenn Sie einen Trail zum Protokollieren von Datenereignissen für alle Amazon S3-Buckets in Ihrem konfigurieren AWS-Konto, ist es einfacher, die CloudTrail Konsole zu verwenden. Informationen zur Verwendung der CloudTrail Konsole zum Konfigurieren eines Trails zum Protokollieren von S3-Datenereignissen finden Sie unter Datenereignisse im AWS CloudTrail -Benutzerhandbuch.

> **Important**
>
> Für Datenereignisse werden zusätzliche Gebühren fällig. Weitere Informationen finden Sie unter AWS CloudTrail – Preise.

Das folgende Verfahren zeigt, wie Sie mit der Amazon S3-Konsole einen CloudTrail Trail zum Protokollieren von Datenereignissen für einen S3-Bucket konfigurieren.

So aktivieren Sie die Protokollierung von CloudTrail Datenereignissen für Objekte in einem S3-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter AWS CloudTrail Datenereignisse die Option Konfigurieren in aus CloudTrail.

Sie können einen neuen CloudTrail Trail erstellen oder einen vorhandenen Trail wiederverwenden und Amazon S3-Datenereignisse so konfigurieren, dass sie in Ihrem Trail protokolliert werden. Informationen zum Erstellen von Trails in der CloudTrail Konsole finden Sie unter Erstellen und Aktualisieren eines Trails mit der Konsole im AWS CloudTrail -Benutzerhandbuch. Informationen zum Konfigurieren der Amazon S3-Datenereignisprotokollierung in der CloudTrail Konsole finden Sie unter Protokollieren von Datenereignissen für Amazon S3-Objekte im AWS CloudTrail -Benutzerhandbuch.

> **Note**
>
> Wenn Sie die CloudTrail Konsole oder die Amazon S3-Konsole verwenden, um einen Trail zum Protokollieren von Datenereignissen für einen S3-Bucket zu konfigurieren, zeigt die Amazon S3-Konsole, dass die Protokollierung auf Objektebene für den Bucket aktiviert ist.

So deaktivieren Sie die Protokollierung von CloudTrail Datenereignissen für Objekte in einem S3-Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die - CloudTrail Konsole unter https://console.aws.amazon.com/cloudtrail/.
2. Wählen Sie im linken Navigationsbereich Trails aus.
3. Wählen Sie den Namen des Trails aus, den Sie erstellt haben, um Ereignisse für den Bucket zu protokollieren.
4. Wählen Sie oben rechts auf der Detailseite des Trails Protokollierung beenden aus.
5. Wählen Sie im anschließend angezeigten Dialogfeld Protokollierung beenden aus.

Weitere Informationen zum Aktivieren der Protokollierung auf Objektebene, wenn Sie einen S3-Bucket erstellen, finden Sie unter Erstellen eines Buckets.

Weitere Informationen zur CloudTrail Protokollierung mit S3-Buckets finden Sie in den folgenden Themen:

- Anzeigen der Eigenschaften eines S3-Buckets
- Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail
- Arbeiten mit CloudTrail Protokolldateien im AWS CloudTrail -Benutzerhandbuch

Identifizieren von Amazon S3-Anforderungen mit CloudTrail

In Amazon S3 können Sie Anforderungen mithilfe eines AWS CloudTrail Ereignisprotokolls identifizieren. AWS CloudTrail ist die bevorzugte Methode zum Identifizieren von Amazon S3-Anforderungen. Wenn Sie jedoch Amazon S3-Serverzugriffsprotokolle verwenden, finden Sie weitere Informationen unter the section called “Identifizieren von S3-Anfragen”.

Themen

Identifizieren von Anforderungen an Amazon S3 in einem CloudTrail Protokoll

Nachdem Sie für CloudTrail die Bereitstellung von Ereignissen an einen Bucket eingerichtet haben, sollten Sie in der Amazon S3-Konsole sehen, wie Objekte in Ihren Ziel-Bucket gelangen. Diese sind folgendermaßen formatiert:

```
s3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/Region/yyyy/mm/dd
```

Ereignisse, die von protokolliert werden, CloudTrail werden als komprimierte gzipped JSON-Objekte in Ihrem S3-Bucket gespeichert. Um Anfragen effizient zu finden, sollten Sie einen Service wie Amazon Athena verwenden, um die CloudTrail Protokolle zu indizieren und abzufragen.

Weitere Informationen zu CloudTrail und Athena finden Sie unter Erstellen der Tabelle für AWS CloudTrail Protokolle in Athena mithilfe der Partitionsprojektion im Amazon Athena-Benutzerhandbuch.

Identifizieren von Anforderungen von Amazon S3 Signature Version 2 mithilfe von CloudTrail

Sie können ein CloudTrail Ereignisprotokoll verwenden, um zu identifizieren, welche API-Signaturversion zum Signieren einer Anforderung in Amazon S3 verwendet wurde. Diese Funktion ist wichtig, weil die Unterstützung für Signature Version 2 deaktiviert wird (veraltet). Danach akzeptiert Amazon S3 keine Anforderungen mit Signature Version 2 mehr, alle Anforderungen müssen also mit Signature Version 4 signiert werden.

Es wird dringend empfohlen, zu verwenden, CloudTrail um festzustellen, ob einer Ihrer Workflows Signature Version 2-Signatur verwendet. Korrigieren Sie diese Workflows, indem Sie die betreffenden

Bibliotheken und den Code so aktualisieren, dass mittels Signature Version 4 signiert wird. Dadurch können Beeinträchtigungen der geschäftlichen Prozesse werden.

Weitere Informationen finden Sie unter Ankündigung: AWS CloudTrail für Amazon S3 fügt neue Felder für verbesserte Sicherheitsüberprüfung hinzu in AWS re:Post.

> **Note**
>
> CloudTrail -Ereignisse für Amazon S3 enthalten die Signaturversion in den Anforderungsdetails unter dem Schlüsselnamen '`additionalEventData`. Um die Signaturversion für Anforderungen zu finden, die an Objekte in Amazon S3 gestellt werden`GET`, z. B. `PUT`-, - und -`DELETE`Anforderungen, müssen Sie CloudTrail Datenereignisse aktivieren. (Diese Option ist standardmäßig deaktiviert.)

AWS CloudTrail ist die bevorzugte Methode zum Identifizieren von Signature Version 2-Anforderungen. Wenn Sie Zugriffsprotokolle von Amazon S3 Server verwenden, lesen Sie Identifizieren von Signature-Version-2-Anforderungen mittels Amazon-S3-Zugriffsprotokollen.

Themen

- Athena-Abfragebeispiele zum Identifizieren von Anforderungen zur Amazon S3 Signature Version 2
- Partitionieren von Signature Version 2-Daten

Athena-Abfragebeispiele zum Identifizieren von Anforderungen zur Amazon S3 Signature Version 2

Example – Wählen Sie alle Signature-Version-2-Ereignisse aus und drucken Sie nur **`EventTime`**, **`S3_Action`**, **`Request_Parameters`**, **`Region`**, **`SourceIP`** und **`UserAgent`**

Ersetzen Sie in der folgenden Athena-Abfrage *`s3_cloudtrail_events_db.cloudtrail_table`* durch Ihre Athena-Details und erhöhen oder entfernen Sie das Limit nach Bedarf.

```
SELECT EventTime, EventName as S3_Action, requestParameters as Request_Parameters,
 awsregion as AWS_Region, sourceipaddress as Source_IP, useragent as User_Agent
FROM s3_cloudtrail_events_db.cloudtrail_table
WHERE eventsource='s3.amazonaws.com'
AND json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
LIMIT 10;
```

Example – Alle Anforderer auswählen, die Signature Version 2-Datenverkehr senden

```
SELECT useridentity.arn, Count(requestid) as RequestCount
FROM s3_cloudtrail_events_db.cloudtrail_table
WHERE eventsource='s3.amazonaws.com'
    and json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
Group by useridentity.arn
```

Partitionieren von Signature Version 2-Daten

Wenn Sie über eine große Datenmenge verfügen, die abgefragt werden muss, können Sie Kosten und Laufzeit für Athena reduzieren, indem Sie eine partitionierte Tabelle erstellen.

Erstellen Sie zu diesem Zweck folgendermaßen eine neue Tabelle mit Partitionen.

```
  CREATE EXTERNAL TABLE s3_cloudtrail_events_db.cloudtrail_table_partitioned(
       eventversion STRING,
       userIdentity STRUCT<
           type:STRING,
           principalid:STRING,
           arn:STRING,
           accountid:STRING,
           invokedby:STRING,
           accesskeyid:STRING,
           userName:STRING,
        sessioncontext:STRUCT<
                   attributes:STRUCT<
                   mfaauthenticated:STRING,
                   creationdate:STRING>,
                   sessionIssuer:STRUCT<
                   type:STRING,
                   principalId:STRING,
                   arn:STRING,
                   accountId:STRING,
                   userName:STRING>
               >
            >,
       eventTime STRING,
       eventSource STRING,
       eventName STRING,
```

```
        awsRegion STRING,
        sourceIpAddress STRING,
        userAgent STRING,
        errorCode STRING,
        errorMessage STRING,
        requestParameters STRING,
        responseElements STRING,
        additionalEventData STRING,
        requestId STRING,
        eventId STRING,
        resources ARRAY<STRUCT<ARN:STRING,accountId: STRING,type:STRING>>,
        eventType STRING,
        apiVersion STRING,
        readOnly STRING,
        recipientAccountId STRING,
        serviceEventDetails STRING,
        sharedEventID STRING,
        vpcEndpointId STRING
    )
    PARTITIONED BY (region string, year string, month string, day string)
    ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.orc.OrcSerde'
    STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
    OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
    LOCATION 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/';
```

Erstellen Sie dann die einzelnen Partitionen. Sie können keine Resultate aus noch nicht erstellten Daten ermitteln.

```
ALTER TABLE s3_cloudtrail_events_db.cloudtrail_table_partitioned ADD
    PARTITION (region= 'us-east-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/us-east-1/2019/02/19/'
    PARTITION (region= 'us-west-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/us-west-1/2019/02/19/'
    PARTITION (region= 'us-west-2', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/us-west-2/2019/02/19/'
    PARTITION (region= 'ap-southeast-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/ap-southeast-1/2019/02/19/'
    PARTITION (region= 'ap-southeast-2', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/ap-southeast-2/2019/02/19/'
    PARTITION (region= 'ap-northeast-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/ap-northeast-1/2019/02/19/'
```

```
    PARTITION (region= 'eu-west-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/eu-west-1/2019/02/19/'
    PARTITION (region= 'sa-east-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/sa-east-1/2019/02/19/';
```

Anschließend können Sie die Anforderung basierend auf diesen Partitionen erstellen und müssen nicht mehr den gesamten Bucket laden.

```
SELECT useridentity.arn,
Count(requestid) AS RequestCount
FROM s3_cloudtrail_events_db.cloudtrail_table_partitioned
WHERE eventsource='s3.amazonaws.com'
AND json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
AND region='us-east-1'
AND year='2019'
AND month='02'
AND day='19'
Group by useridentity.arn
```

Identifizieren des Zugriffs auf S3-Objekte mithilfe von CloudTrail

Sie können Ihre AWS CloudTrail Ereignisprotokolle verwenden, um Amazon S3-Objektzugriffsanforderungen für Datenereignisse wie `GetObject`, `DeleteObject`und zu identifizieren `PutObject`und zusätzliche Informationen zu diesen Anforderungen zu entdecken.

Das folgende Beispiel zeigt, wie alle PUT Objektanforderungen für Amazon S3 aus einem AWS CloudTrail Ereignisprotokoll abgerufen werden.

Themen

- Athena-Abfragebeispiele zur Identifizierung von Amazon-S3-Objektzugriffsanfragen

Athena-Abfragebeispiele zur Identifizierung von Amazon-S3-Objektzugriffsanfragen

Ersetzen Sie in den folgenden Athena-Beispielabfragen *s3_cloudtrail_events_db.cloudtrail_table* durch Ihre Athena-Details und ändern Sie den Datumsbereich nach Bedarf.

Example – Wählen Sie alle Ereignisse aus, für die **PUT**-Objektzugriffsanforderungen vorliegen, und drucken Sie nur **EventTime**, **EventSource**, **SourceIP**, **UserAgent**, **BucketName**, **object** und **UserARN**

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  json_extract_scalar(requestParameters, '$.key') as object,
  userIdentity.arn as userArn
FROM
  s3_cloudtrail_events_db.cloudtrail_table
WHERE
  eventName = 'PutObject'
  AND eventTime BETWEEN '2019-07-05T00:00:00Z' and '2019-07-06T00:00:00Z'
```

Example – Wählen Sie alle Ereignisse aus, für die **GET**-Objektzugriffsanforderungen vorliegen, und drucken Sie nur **EventTime**, **EventSource**, **SourceIP**, **UserAgent**, **BucketName**, **object** und **UserARN**

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  json_extract_scalar(requestParameters, '$.key') as object,
  userIdentity.arn as userArn
FROM
  s3_cloudtrail_events_db.cloudtrail_table
WHERE
  eventName = 'GetObject'
  AND eventTime BETWEEN '2019-07-05T00:00:00Z' and '2019-07-06T00:00:00Z'
```

Example – Wählen Sie alle anonymen Anforderungsereignisse für einen Bucket in einem bestimmten Zeitraum aus und drucken Sie nur **`EventTime`**, **`EventName`**, **`EventSource`**, **`SourceIP`**, **`UserAgent`**, **`BucketName`**, **`UserARN`** und **`AccountID`**

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  userIdentity.arn as userArn,
  userIdentity.accountId
FROM
  s3_cloudtrail_events_db.cloudtrail_table
WHERE
  userIdentity.accountId = 'anonymous'
  AND eventTime BETWEEN '2019-07-05T00:00:00Z' and '2019-07-06T00:00:00Z'
```

Example – Identifizieren Sie alle Anforderungen, für die eine ACL zur Autorisierung erforderlich war

Das folgende Amazon-Athena-Abfragebeispiel zeigt, wie alle Anforderungen für Ihre S3-Buckets identifiziert werden, für die eine Zugriffssteuerungsliste (ACL) zur Autorisierung erforderlich war. Wenn für die Anforderung eine ACL zur Autorisierung erforderlich war, lautet der `aclRequired`-Wert in `additionalEventData` `Yes`. Wenn keine ACLs erforderlich waren, ist `aclRequired` nicht vorhanden. Sie können diese Informationen verwenden, um diese ACL-Berechtigungen zu den entsprechenden Bucket-Richtlinien zu migrieren. Nachdem Sie diese Bucket-Richtlinien erstellt haben, können Sie ACLs für diese Buckets deaktivieren. Weitere Informationen über das Deaktivieren von ACLs finden Sie unter Voraussetzungen für die Deaktivierung von ACLs.

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  userIdentity.arn as userArn,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  json_extract_scalar(requestParameters, '$.key') as object,
  json_extract_scalar(additionalEventData, '$.aclRequired') as aclRequired
FROM
```

```
  s3_cloudtrail_events_db.cloudtrail_table
WHERE
  json_extract_scalar(additionalEventData, '$.aclRequired') = 'Yes'
  AND eventTime BETWEEN '2022-05-10T00:00:00Z' and '2022-08-10T00:00:00Z'
```

> **Note**
>
> - Diese Abfragebeispiele können auch für die Sicherheitsüberwachung nützlich sein. Sie können die Ergebnisse auf `PutObject`- oder `GetObject`-Aufrufe von unerwarteten oder nicht autorisierten IP-Adressen oder Anforderern und zum Identifizieren anonymer Anforderungen an Ihre Buckets prüfen.
> - Diese Abfrage ruft nur Informationen von der Zeit ab, zu der die Protokollierung aktiviert wurde.

Wenn Sie Amazon-S3-Server-Zugriffsprotokolle verwenden, lesen Sie Identifizieren von Objektzugriffsanforderungen mittels Amazon-S3-Zugriffsprotokollen.

Protokollieren von Anfragen mit Server-Zugriffsprotokollierung

Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anforderungen, die an einen Bucket gestellt wurden. Server-Zugriffsprotokolle sind für viele Anwendungen nützlich. Beispielsweise können Zugriffsprotokoll-Informationen bei Sicherheits- und Zugriffsprüfungen nützlich sein. Diese Informationen können Ihnen auch helfen, mehr über Ihre Kundenbasis zu erfahren und Ihre Amazon-S3-Rechnung zu verstehen.

> **Note**
>
> Serverzugriffsprotokolle protokollieren keine Informationen zu Fehlern wegen Umleitungen in falsche Regionen, die nach dem 20. März 2019 gestartet wurden. Fehler wegen Umleitungen in falsche Regionen treten auf, wenn eine Anforderung für ein Objekt oder einen Bucket außerhalb der Region gesendet wird, in der sich der Bucket befindet.

Wie aktiviere ich die Protokollzustellung?

Um die Protokollbereitstellung zu aktivieren, führen Sie die folgenden grundlegenden Schritte aus. Details hierzu finden Sie unter Aktivieren Sie die Amazon-S3-Server-Zugriffsprotokollierung.

1. Geben Sie den Namen des Ziel-Buckets an an (auch als Ziel-Bucket bezeichnet). In diesem Bucket soll Amazon S3 die Zugriffsprotokolle als Objekte speichern. Quell- und Ziel-Bucket müssen sich in derselben AWS-Region befinden und demselben Konto gehören. Der Ziel-Bucket darf keine S3-Object-Lock-Standardkonfiguration für die Aufbewahrungsdauer besitzen. Für den Ziel-Bucket darf außerdem die Option „Zahlung durch den Anforderer“ nicht aktiviert sein.

   Sie können Protokolle in jeden Bucket speichern lassen, der sich in der gleichen Region wie der Quell-Bucket befindet, einschließlich des Quell-Buckets selbst. Zur einfacheren Protokollverwaltung empfehlen wir jedoch, Zugriffsprotokolle in einem anderen Bucket zu speichern.

   Wenn Quell- und Ziel-Bucket identisch sind, werden zusätzliche Protokolle für die Protokolle erstellt, die zum Bucket geschrieben werden. Hierdurch entsteht eine Endlosprotokollschleife. Diese Vorgehensweise wird nicht empfohlen, da sie zu einer geringfügigen Erhöhung der Speicherkosten führen könnte. Weiterhin könnten die zusätzlichen Protokolle über Protokolle das Auffinden des gesuchten Protokolls erschweren.

   Wenn Sie Zugriffsprotokolle im Quell-Bucket speichern möchten, sollten Sie ein Zielpräfix (Zielpräfix) für alle Protokollobjektschlüssel angeben. Wenn Sie ein Präfix angeben, beginnen alle Protokollobjektnamen mit einer gemeinsamen Zeichenfolge. So können Sie die Protokollobjekte leichter identifizieren.

2. (Optional) Weisen Sie allen Amazon-S3-Protokollobjektschlüsseln ein Zielpräfix zu. Das Zielpräfix (Zielpräfix) macht es einfacher für Sie, die Protokollobjekte zu finden. Wenn Sie beispielsweise den Präfixwert `logs/` angeben, beginnt jedes von Amazon S3 erstellte Protokollobjekt mit dem Präfix `logs/` im Schlüssel, z. B.:

   ```
   logs/2013-11-01-21-32-16-E568B2907131C0C0
   ```

   Wenn Sie den Präfixwert `logs` angeben, sieht das Protokollobjekt wie folgt aus:

   ```
   logs2013-11-01-21-32-16-E568B2907131C0C0
   ```

   Präfixe sind auch nützlich, um zwischen Quell-Buckets zu unterscheiden, wenn mehrere Buckets zum selben Ziel-Bucket protokolliert werden.

   Dieses Präfix ist auch beim Löschen der Protokolle nützlich. Beispielsweise können Sie eine Lebenszyklus-Konfigurationsregel für Amazon S3 festlegen, um Objekte mit einem

bestimmten Präfix zu löschen. Weitere Informationen finden Sie unter Löschen von Amazon-S3-Protokolldateien.

3. (Optional) Legen Sie Berechtigungen fest, um anderen Benutzern Zugriff auf die generierten Protokolle zu gewähren. Standardmäßig hat nur der Bucket-Eigentümer stets vollen Zugriff auf die Protokollobjekte. Wenn Ihr Ziel-Bucket die Einstellung „Von Bucket-Besitzer erzwungen“ für S3 Object Ownership festlegt, um Zugriffssteuerungslisten (ACLs) zu deaktivieren, können Sie keine Berechtigungen in Zielgewährungen (Zielgewährungen) gewähren, die ACLs verwenden. Sie können jedoch Ihre Bucket-Richtlinie für den Ziel-Bucket aktualisieren, um anderen Benutzern Zugriff zu gewähren. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3 und Berechtigungen für Protokollbereitstellung.
4. (Optional) Legen Sie ein Protokollobjekt-Schlüsselformat für die Protokolldateien fest. Es gibt zwei Optionen für das Protokollobjekt-Schlüsselformat (auch als Zielobjekt-Schlüsselformat bezeichnet):
   - N-on-date-based Partitionierung – Dies ist das ursprüngliche Protokollobjekt-Schlüsselformat. Wenn Sie dieses Format wählen, sieht das Protokolldatei-Schlüsselformat wie folgt aus:

     ```
     [DestinationPrefix][YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
     ```

     Wenn Sie beispielsweise `logs/` als Präfix angeben, werden Ihre Protokollobjekte wie folgt benannt:

     ```
     logs/2013-11-01-21-32-16-E568B2907131C0C0
     ```

   - Partitionierung mit Datum – Wenn Sie die Partitionierung mit Datum wählen, können Sie die Ereignis- oder Bereitstellungszeit für die Protokolldatei als die Datumsquelle wählen, die im Protokollformat verwendet wird. Dieses Format vereinfacht das Abfragen der Protokolle.

     Wenn Sie die Partitionierung mit Datum wählen, sieht das Protokolldatei-Schlüsselformat wie folgt aus:

     ```
     [DestinationPrefix][SourceAccountId]/[SourceRegion]/[SourceBucket]/[YYYY]/[MM]/
     [DD]/[YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
     ```

     Wenn Sie beispielsweise `logs/` als Zielpräfix angeben, werden Ihre Protokollobjekte wie folgt benannt:

     ```
     logs/123456789012/us-west-2/DOC-EXAMPLE-SOURCE-
     BUCKET/2023/03/01/2023-03-01-21-32-16-E568B2907131C0C0
     ```

Bei der Bereitstellung nach Bereitstellungszeit entspricht die Zeit in den Namen der Protokolldateien der Bereitstellungszeit für die Protokolldateien.

Bei der Bereitstellung nach Ereigniszeit entsprechen die Angaben für Jahr, Monat und Tag dem Tag, an dem das Ereignis eingetreten ist. Stunde, Minuten und Sekunden sind im Schlüssel auf `00` festgelegt. Die in diesen Protokolldateien bereitgestellten Protokolle beziehen sich nur auf einen bestimmten Tag.

Wenn Sie Ihre Protokolle über die AWS Command Line Interface (AWS CLI), AWS SDKs oder die Amazon-S3-REST-API konfigurieren, verwenden Sie , `TargetObjectKeyFormat` um das Protokollobjekt-Schlüsselformat anzugeben. Amazon S3 Um die non-date-based Partitionierung anzugeben, verwenden Sie `SimplePrefix`. Um eine Partitionierung mit Datum anzugeben, verwenden Sie `PartitionedPrefix`. Bei Verwendung von `PartitionedPrefix` verwenden Sie `PartitionDateSource`, um `EventTime` oder `DeliveryTime` anzugeben.

Für `SimplePrefix` sieht das Protokolldatei-Schlüsselformat wie folgt aus:

```
[TargetPrefix][YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
```

Für `PartitionedPrefix` mit Ereignis- oder Bereitstellungszeit sieht das Protokolldatei-Schlüsselformat wie folgt aus:

```
[TargetPrefix][SourceAccountId]/[SourceRegion]/[SourceBucket]/[YYYY]/[MM]/[DD]/
[YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
```

Protokollobjekt-Schlüsselformat

Amazon S3 verwendet die folgenden Objektschlüsselformate für die Protokollobjekte, die in den Ziel-Bucket hochgeladen werden:

- N-on-date-based Partitionierung – Dies ist das ursprüngliche Protokollobjekt-Schlüsselformat. Wenn Sie dieses Format wählen, sieht das Protokolldatei-Schlüsselformat wie folgt aus:

```
[DestinationPrefix][YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
```

- Partitionierung mit Datum – Wenn Sie die Partitionierung mit Datum wählen, können Sie die Ereignis- oder Bereitstellungszeit für die Protokolldatei als die Datumsquelle wählen, die im Protokollformat verwendet wird. Dieses Format vereinfacht das Abfragen der Protokolle.

  Wenn Sie die Partitionierung mit Datum wählen, sieht das Protokolldatei-Schlüsselformat wie folgt aus:

```
[DestinationPrefix][SourceAccountId]/[SourceRegion]/[SourceBucket]/[YYYY]/[MM]/[DD]/
[YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
```

Im Protokollobjektschlüssel sind YYYY, `MM`, `DD`, `hh`, `mm` und `ss` jeweils die Stellen für Jahr, Monat, Tag, Stunde, Minute und Sekunden. Datum und Uhrzeit entsprechen der Zeitzone UTC (Coordinated Universal Time).

Eine Protokolldatei, die zu einem bestimmten Zeitpunkt bereitgestellt wurde, kann Datensätze enthalten, die an einem beliebigen Zeitpunkt davor geschrieben wurden. Es lässt sich nicht feststellen, ob alle Protokoll-Datensätze für ein bestimmtes Zeitintervall bereitgestellt wurden oder nicht.

Die `UniqueString`-Komponente des Schlüssels verhindert, dass Dateien überschrieben werden. Sie hat keine Bedeutung und wird normalerweise von Protokollverarbeitungssoftware ignoriert.

Wie werden Protokolle ausgeliefert?

Amazon S3 sammelt regelmäßig Zugriffsprotokoll-Datensätze, konsolidiert die Datensätze in Protokolldateien und lädt anschließend die Protokolldateien als Protokollobjekte zu Ihrem Ziel-Bucket hoch. Wenn Sie die Protokollierung für mehrere Quell-Buckets mit demselben Ziel-Bucket aktivieren, besitzt der Ziel-Bucket Zugriffsprotokolle für alle diese Quell-Buckets. Jedes Protokollobjekt gibt jedoch Zugriffsprotokoll-Datensätze für einen bestimmten Quell-Bucket aus.

Amazon S3 verwendet ein spezielles Protokollbereitstellungskonto zum Schreiben der Server-Zugriffsprotokolle. Für diese Protokolle gelten die normalen Zugriffskontrollbeschränkungen. Wir empfehlen Ihnen, die Bucket-Richtlinie für den Ziel-Bucket zu aktualisieren, um Zugriff auf den Protokollservice-Prinzipal (`logging.s3.amazonaws.com`) zur Zugriffsprotokollbereitstellung zu gewähren. Sie können über Ihre Bucket-Zugriffssteuerungsliste (ACL) der S3-Protokollbereitstellungsgruppe auch Zugriff für die Zugriffsprotokollbereitstellung gewähren. Die Gewährung des Zugriffs für die S3-Protokollbereitstellungsgruppe über Ihre Bucket-ACL wird jedoch nicht empfohlen.

Wenn Sie die Serverzugriffsprotokollierung über die Ziel-Bucket-Richtlinie aktivieren und Zugriff für die Zugriffsprotokollbereitstellung gewähren, müssen Sie die Richtlinie aktualisieren, um `s3:PutObject`-Zugriff für den Protokollierungsservice-Prinzipal zuzulassen. Wenn Sie die Serverzugriffsprotokollierung über die Amazon-S3-Konsole aktivieren, aktualisiert die Konsole die Ziel-Bucket-Richtlinie automatisch, um dem Protokollierungsservice-Prinzipal diese Berechtigungen zu gewähren. Weitere Informationen zum Erteilen von Berechtigungen für die Zustellung von Serverzugriffsprotokollen finden Sie unter Berechtigungen für Protokollbereitstellung.

> **Note**
>
> Bei Virtual-Private-Cloud (VPC)-Endpunktanforderungen werden dem Anforderer oder Bucket-Besitzer keine Serverzugriffsprotokolle bereitgestellt, wenn die VPC-Endpunktrichtlinie solche Anforderungen nicht zulässt.

Vom Bucket-Eigentümer erzwungene Einstellung für S3 Object Ownership

Wenn der Ziel-Bucket die Einstellung „Von Bucket-Besitzer erzwungen“ für Object Ownership verwendet, sind ACLs deaktiviert und wirken sich nicht mehr auf Berechtigungen aus. Sie müssen die Bucket-Richtlinie aktualisieren, damit der Ziel-Bucket Zugriff auf den Protokollierungsservice-Prinzipal gewähren kann. Informationen zu Object Ownership finden Sie unter Gewähren von Zugriff auf die S3-Protokollbereitstellungsgruppe für die Protokollierung des Serverzugriffs.

Best-Effort-Serverprotokollbereitstellung

Serverzugriffsprotokoll-Datensätze werden auf Best-Effort-Grundlage bereitgestellt. Die meisten Anforderungen nach einem Bucket, der für die Protokollierung richtig konfiguriert ist, führen zu einem ausgelieferten Protokollsatz. Die meisten Protokollsätze werden innerhalb weniger Stunden nach der Aufnahme geliefert, können aber häufiger geliefert werden.

Die Vollständigkeit und Aktualität der Serverprotokollierung wird nicht garantiert. Der Protokolldatensatz für eine bestimmte Anforderung wird möglicherweise viel später bereitgestellt, als die Anforderung tatsächlich verarbeitet wurde; es kann auch sein, dass er gar nicht bereitgestellt wird. Es ist möglich, dass Sie sogar ein Duplikat eines Protokolldatensatzes sehen. Der Zweck der Serverprotokolle besteht darin, Ihnen einen Überblick über die Art des Datenverkehrs zu und von Ihrem Bucket zu vermitteln. Auch wenn Protokolldatensätze selten verloren gehen oder dupliziert werden, sollten Sie wissen, dass die Serverprotokollierung nicht der vollständigen Erfassung aller Anforderungen dient.

Aufgrund der Best-Effort-Grundlage der Serverprotokollierung enthalten Ihre Nutzungsberichte möglicherweise eine oder mehrere Zugriffsanforderungen, die nicht in einem bereitgestellten Serverprotokoll angezeigt werden. Sie finden diese Nutzungsberichte unter Kosten- und Nutzungsberichte in der AWS Billing and Cost Management -Konsole.

Statusänderungen in der Bucket-Protokollierung werden mit der Zeit wirksam

Änderungen am Protokollierungsstatus eines Buckets benötigen einige Zeit, bis sie sich auf die Bereitstellung von Protokolldateien auswirken. Wenn Sie die Protokollierung für einen Bucket aktivieren, werden in der folgenden Stunde möglicherweise einige Anforderungen protokolliert und andere nicht. Angenommen, Sie ändern den Ziel-Bucket für die Protokollierung von Bucket A in Bucket B. In der nächsten Stunde werden möglicherweise einige Protokolle in Bucket A bereitgestellt, während andere Protokolle im neuen Ziel-Bucket B bereitgestellt werden. In jedem Fall werden die neuen Einstellungen letztendlich ohne weiteres Eingreifen Ihrerseits wirksam.

Weitere Informationen zu Protokollierungs- und Protokolldateien finden Sie in den folgenden Abschnitten:

Themen

Aktivieren Sie die Amazon-S3-Server-Zugriffsprotokollierung

Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anfragen, die an einen Amazon-S3-Bucket gestellt wurden. Server-Zugriffsprotokolle sind für viele Anwendungen nützlich. Beispielsweise können Zugriffsprotokoll-Informationen bei Sicherheits- und Zugriffsprüfungen nützlich sein. Diese Informationen können Ihnen auch helfen, mehr über Ihre Kundenbasis zu erfahren und Ihre Amazon-S3-Rechnung zu verstehen.

Standardmäßig erfasst Amazon S3 keine Server-Zugriffsprotokolle. Wenn Sie die Protokollierung aktivieren, stellt Amazon S3 Zugriffsprotokolle für einen Quell-Bucket in einem von Ihnen ausgewählten Ziel-Bucket (Ziel-Bucket) bereit. Der Ziel-Bucket muss sich in derselben AWS-Region und im selben AWS-Konto wie der Ziel-Bucket befinden.

Ein Zugriffsprotokollsatz enthält Details über die Anforderungen, die an einen Bucket gestellt werden. Dabei kann es sich um den Anforderungstyp, die in der Anfrage angegebenen Ressourcen sowie Uhrzeit und Datum der Anfrageverarbeitung handeln. Weitere Informationen zu den Grundlagen der Protokollierung finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

Important

- Für die Aktivierung der Server-Zugriffsprotokollierung auf einem Amazon-S3-Bucket fallen keine zusätzlichen Kosten an. Für die Speicherung der Protokolldateien, die das System an Sie überträgt, fallen allerdings die normalen Gebühren an. (Sie können die Protokolldateien jederzeit löschen.) Wir berechnen keine Datenübertragungskosten für die Bereitstellung von Protokolldateien. Für den Zugriff auf die Protokolldateien fallen jedoch die regulären Kosten für Datenübertragungen an.
- Für Ihren Ziel-Bucket sollte keine Serverzugriffsprotokollierung aktiviert sein. Sie können Protokolle in jeden Bucket speichern lassen, der sich in der gleichen Region wie der Quell-Bucket befindet, einschließlich des Quell-Buckets selbst. Die Zustellung von Protokollen an den Quell-Bucket führt jedoch zu einer unendlichen Schleife von Protokollen und wird nicht empfohlen. Zur einfacheren Protokollverwaltung empfehlen wir, Zugriffsprotokolle in einem anderen Bucket zu speichern. Weitere Informationen finden Sie unter Wie aktiviere ich die Protokollzustellung?.
- S3-Buckets, für die S3 Object Lock aktiviert ist, können nicht als Ziel-Buckets für Serverzugriffsprotokolle verwendet werden. Ihr Ziel-Bucket darf nicht für eine Standardaufbewahrungsdauer konfiguriert sein.
- Für den Ziel-Bucket darf die Option „Zahlung durch den Anforderer“ nicht aktiviert sein.
- Sie können für den Ziel-Bucket die Bucket-Standardverschlüsselung nur dann verwenden, wenn Sie die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) nutzen, die den 256-Bit Advanced Encryption Standard (AES-256) verwenden. Die serverseitige Standardverschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) wird nicht unterstützt.

Sie können die Protokollierung des Serverzugriffs mithilfe der Amazon-S3-Konsole, der Amazon-S3-API, des AWS Command Line Interface (AWS CLI) oder AWS -SDKs aktivieren oder deaktivieren.

Berechtigungen für Protokollbereitstellung

Amazon S3 verwendet ein spezielles Protokollbereitstellungskonto zum Schreiben der Server-Zugriffsprotokolle. Für diese Protokolle gelten die normalen Zugriffskontrollbeschränkungen. Zur Bereitstellung von Zugriffsprotokollen müssen Sie dem Protokollierungsservice-Prinzipal (`logging.s3.amazonaws.com`) Zugriff auf den Ziel-Bucket gewähren.

Zur Gewährung von Berechtigungen für die Protokollbereitstellung für Amazon S3 können Sie eine Bucket-Richtlinie oder Bucket-Zugriffssteuerungslisten (ACLs) verwenden, abhängig von den S3-Object-Ownership-Einstellungen für den Ziel-Bucket. Wir empfehlen jedoch die Empfehlung einer Bucket-Richtlinie anstelle von ACLs.

Vom Bucket-Eigentümer erzwungene Einstellung für S3 Object Ownership

Wenn der Ziel-Bucket die Einstellung „Von Bucket-Besitzer erzwungen“ für Object Ownership verwendet, sind ACLs deaktiviert und wirken sich nicht mehr auf Berechtigungen aus. In diesem Fall müssen Sie die Bucket-Richtlinie für den Ziel-Bucket aktualisieren, um Zugriff auf den Protokollierungsservice-Prinzipal zu gewähren. Sie können Ihre Bucket-ACL nicht aktualisieren, um Zugriff auf die S3-Protokollbereitstellungsgruppe zu gewähren. Sie können auch keine Zielgewährungen (Zielgewährungen) in Ihre PutBucketLogging-Konfiguration einfügen.

Informationen zum Migrieren vorhandener Bucket-ACLs für die Zustellung von Zugriffsprotokollen auf eine Bucket-Richtlinie finden Sie unter Gewähren von Zugriff auf die S3-Protokollbereitstellungsgruppe für die Protokollierung des Serverzugriffs. Informationen zu Object Ownership finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket.. Wenn Sie neue Buckets erstellen, sind ACLs standardmäßig deaktiviert.

Gewähren von Zugriff über eine Bucket-Richtlinie

Um über die Bucket-Richtlinie Zugriff auf den Ziel-Bucket zu gewähren, aktualisieren Sie die Bucket-Richtlinie so, dass sie die `s3:PutObject`-Berechtigung für den Protokollierungsservice-Prinzipal zulässt. Wenn Sie die Serverzugriffsprotokollierung über die Amazon-S3-Konsole aktivieren, aktualisiert die Konsole die Bucket-Richtlinie für den Ziel-Bucket automatisch, um dem Protokollierungsservice-Prinzipal diese Berechtigung zu gewähren. Wenn Sie die Serverzugriffsprotokollierung programmgesteuert aktivieren, müssen Sie die Bucket-Richtlinie für den Ziel-Bucket manuell aktualisieren, um Zugriff auf den Protokollierungsservice-Prinzipal zu gewähren.

Ein Beispiel für eine Bucket-Richtlinie, die Zugriff auf den Protokollierungsservice-Prinzipal gewährt, finden Sie unter the section called “Erteilen von Berechtigungen für den Prinzipal des Protokollierungsservices mithilfe einer Bucket-Richtlinie”.

Gewähren von Zugriff über Bucket-ACLs

Sie können abwechselnd Bucket-ACLs verwenden, um Zugriff auf die Zugriffsprotokollbereitstellung zu gewähren. Sie fügen der Bucket-ACL einen Erteilungseintrag hinzu, der `WRITE`-und `READ_ACP`-Berechtigungen für die S3-Protokollbereitstellungs-Gruppe gewährt. Es wird jedoch nicht empfohlen, den Zugriff auf die S3-Protokollbereitstellungsgruppe mit Ihren Bucket-ACLs zu gewähren. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket.. Informationen zum Migrieren vorhandener Bucket-ACLs für die Zustellung von Zugriffsprotokollen auf eine Bucket-Richtlinie finden Sie unter Gewähren von Zugriff auf die S3-Protokollbereitstellungsgruppe für die Protokollierung des Serverzugriffs. Ein Beispiel für eine ACL, die Zugriff auf den Protokollierungsservice-Prinzipal gewährt, finden Sie unter the section called “Erteilen von Berechtigungen für die Protokollbereitstellungsgruppe mithilfe einer Bucket-ACL”.

Erteilen von Berechtigungen für den Prinzipal des Protokollierungsservices mithilfe einer Bucket-Richtlinie

Diese Bucket-Beispielrichtlinie gewährt dem Protokollierungsservice-Prinzipal (`logging.s3.amazonaws.com`) die `s3:PutObject`-Berechtigung. Wenn Sie diese Bucket-Richtlinie verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen. In der folgenden Richtlinie *`DOC-EXAMPLE-DESTINATION-BUCKET`* ist der Ziel-Bucket, an den Serverzugriffsprotokolle geliefert werden, und *`DOC-EXAMPLE-SOURCE-BUCKET`* ist der Quell-Bucket. *`EXAMPLE-LOGGING-PREFIX`* ist das optionale Zielpräfix (auch als Zielpräfix bezeichnet), das Sie für Ihre Protokollobjekte verwenden möchten. *`SOURCE-ACCOUNT-ID`* ist der AWS-Konto , der den Quell-Bucket besitzt.

> **Note**
>
> Wenn Ihre Bucket-Richtlinie `Deny`-Anweisungen enthält, stellen Sie sicher, dass diese Anweisungen Amazon S3 nicht daran hindern, Zugriffsprotokolle bereitzustellen.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "S3ServerAccessLogsPolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/EXAMPLE-LOGGING-
PREFIX*",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
                },
                "StringEquals": {
                    "aws:SourceAccount": "SOURCE-ACCOUNT-ID"
                }
            }
        }
    ]
}
```

Erteilen von Berechtigungen für die Protokollbereitstellungsgruppe mithilfe einer Bucket-ACL

> **Note**
>
> Als bewährte Sicherheitsmethode deaktiviert Amazon S3 standardmäßig Zugriffssteuerungslisten (ACLs) in allen neuen Buckets. Weitere Informationen zu ACL-Berechtigungen in der Amazon-S3-Konsole finden Sie unter Konfigurieren von ACLs.

Auch wenn wir diesen Ansatz nicht empfehlen, können Sie der Protokollbereitstellungsgruppe über eine Bucket-ACL Berechtigungen gewähren. Wenn der Ziel-Bucket jedoch die Einstellung „Von Bucket-Besitzer erzwungen“ für Object Ownership verwendet, können Sie keine Bucket- oder Objekt-ACLs festlegen. Sie können auch keine Zielgewährungen (Zielgewährungen) in Ihre PutBucketLogging-Konfiguration einfügen. Stattdessen müssen Sie eine Bucket-Richtlinie verwenden, um dem Prinzipal des Protokollierungsservices (`logging.s3.amazonaws.com`) Zugriff zu gewähren. Weitere Informationen finden Sie unter Berechtigungen für Protokollbereitstellung.

In der Bucket-ACL wird die Protokollbereitstellungsgruppe durch die folgende URL dargestellt:

```
http://acs.amazonaws.com/groups/s3/LogDelivery
```

Um `WRITE`- und `READ_ACP`- (ACL-Read)-Berechtigungen zu gewähren, fügen Sie der Ziel-Bucket-ACL die folgenden Gewährungen hinzu:

```
<Grant>
    <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
    </Grantee>
    <Permission>WRITE</Permission>
</Grant>
<Grant>
    <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
    </Grantee>
    <Permission>READ_ACP</Permission>
</Grant>
```

Beispiele zum Hinzufügen von programmgesteuerten ACL-Berechtigungen finden Sie unter Konfigurieren von ACLs.

> **Important**
>
> Wenn Sie die Amazon S3-Serverzugriffsprotokollierung mithilfe von AWS CloudFormation für einen Bucket aktivieren und ACLs verwenden, um Zugriff auf die S3-Protokollbereitstellungsgruppe zu gewähren, müssen Sie Ihrer CloudFormation Vorlage auch hinzufügen`AccessControl": "LogDeliveryWrite"`. Dies ist wichtig, da Sie diese Berechtigungen nur erteilen können, indem Sie eine ACL für den Bucket erstellen, aber Sie können keine benutzerdefinierten ACLs für Buckets in erstellen CloudFormation. Sie können nur vordefinierte ACLs mit verwenden CloudFormation.

Vorgehensweise zum Aktivieren der Server-Zugriffsprotokollierung

AWS CLI Gehen Sie wie folgt vor, um die Serverzugriffsprotokollierung mithilfe der Amazon S3-Konsole, der Amazon S3-REST-API, der - AWS SDKs und zu aktivieren. SDKs

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie die Server-Zugriffsprotokollierung aktivieren möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie im Abschnitt Server access logging (Server-Zugriffsprotokollierung) die Option Edit (Bearbeiten) aus.
5. Wählen Sie unter Serverzugriffsprotokollierung) die Option Aktivieren aus.
6. Geben Sie unter Ziel-Bucket einen Bucket und ein optionales Präfix an. Wenn Sie ein Präfix angeben, sollten Sie nach dem Präfix einen Schrägstrich (/) einfügen, damit Sie Ihre Protokolle leichter finden können.

   Note

   Wenn Sie ein Präfix mit einem Schrägstrich (/) angeben, können Sie die Protokollobjekte leichter finden. Wenn Sie beispielsweise den Präfixwert `logs/` angeben, beginnt jedes von Amazon S3 erstellte Protokollobjekt mit dem Präfix `logs/` im Schlüssel wie folgt:

   ```
   logs/2013-11-01-21-32-16-E568B2907131C0C0
   ```

   Wenn Sie den Präfixwert `logs` angeben, sieht das Protokollobjekt wie folgt aus:

   ```
   logs2013-11-01-21-32-16-E568B2907131C0C0
   ```

7. Führen Sie unter Protokollobjekt-Schlüsselformat eine der folgenden Aktionen aus:

   - Um die non-date-based Partitionierung auszuwählen, wählen Sie [DestinationPrefixJJJJ]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString] aus.
   - Um die datumsbasierte Partitionierung auszuwählen, wählen Sie [DestinationPrefixBolSourceAccountId]/[SourceRegion]/[SourceBucket]/[JJJJ]/[MM]/[DD]/[YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString] und dann S3-Ereigniszeit oder Protokolldatei-Bereitstellungszeit aus.

8. Wählen Sie Änderungen speichern aus.

Wenn Sie die Serverzugriffsprotokollierung für einen Bucket aktivieren, aktiviert die Konsole die Protokollierung für den Quell-Bucket und aktualisiert die Bucket-Richtlinie für den Ziel-Bucket, um dem Protokollierungsservice-Prinzipal (`logging.s3.amazonaws.com`) die `s3:PutObject`-Berechtigung zu gewähren. Weitere Informationen zu dieser Bucket-Richtlinie finden Sie unter Erteilen von Berechtigungen für den Prinzipal des Protokollierungsservices mithilfe einer Bucket-Richtlinie.

Sie können die Protokolle im Ziel-Bucket anzeigen. Nachdem Sie die Server-Zugriffsprotokollierung aktiviert haben, kann es einige Stunden dauern, bis die Protokolle in den Ziel-Bucket geliefert werden. Weitere Informationen darüber, wie und wann Protokolle bereitgestellt werden, finden Sie unter Wie werden Protokolle ausgeliefert?.

Weitere Informationen finden Sie unter Anzeigen der Eigenschaften eines S3-Buckets.

Verwenden der REST-API

Zum Aktivieren der Protokollierung senden Sie eine PutBucketLogging-Anforderung, um die Protokollierungskonfiguration zum Quell-Bucket hinzuzufügen. Die Anforderung gibt den Ziel-Bucket (Ziel-Bucket) und optional das Präfix an, das für alle Protokollobjektschlüssel verwendet werden soll.

Das folgende Beispiel gibt *`DOC-EXAMPLE-DESTINATION-BUCKET`* als Ziel-Bucket und *`logs/`* als Präfix an.

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>DOC-EXAMPLE-DESTINATION-BUCKET</TargetBucket>
    <TargetPrefix>logs/</TargetPrefix>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Das folgende Beispiel gibt *`DOC-EXAMPLE-DESTINATION-BUCKET`* als Ziel-Bucket, *`logs/`* als Präfix und `EventTime` als Protokollobjekt-Schlüsselformat an.

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>DOC-EXAMPLE-DESTINATION-BUCKET</TargetBucket>
    <TargetPrefix>logs/</TargetPrefix>
    <TargetObjectKeyFormat>
      <PartitionedPrefix>
```

```
        <PartitionDateSource>EventTime</PartitionDateSource>
      </PartitionedPrefix>
  </TargetObjectKeyFormat>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Die Protokollobjekte werden vom S3-Protokollübermittlungskonto geschrieben und besitzen dessen Eigentümer, und der Bucket-Eigentümer erhält vollständige Berechtigungen für die Protokollobjekte. Sie können optional Zielgewährungen verwenden (Zielgewährungen), um anderen Benutzern Berechtigungen zu gewähren, damit diese auf die Protokolle zugreifen können. Weitere Informationen finden Sie unter PutBucketLogging.

> **Note**
>
> Wenn der Ziel-Bucket die Einstellung „Von Bucket-Besitzer erzwungen“ für Object Ownership verwendet, können Sie keine Zielgewährungen verwenden, um anderen Benutzern Berechtigungen zu gewähren. Um anderen Benutzern Berechtigungen zu gewähren, können Sie die Bucket-Richtlinie für den Ziel-Bucket aktualisieren. Weitere Informationen finden Sie unter Berechtigungen für Protokollbereitstellung.

Zum Abruf der Protokollierungskonfiguration für einen Bucket verwenden Sie die API-Operation GetBucketLogging.

Um die Protokollierungskonfiguration zu löschen, senden Sie eine `PutBucketLogging`-Anforderung mit einem leeren `BucketLoggingStatus`:

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
</BucketLoggingStatus>
```

Um die Protokollierung für einen Bucket zu aktivieren, können Sie entweder die Amazon S3-API oder die AWS -SDK-Wrapper-Bibliotheken verwenden.

Verwenden der AWS SDKs

In den folgenden Beispielen wird die Protokollierung für einen Bucket aktiviert. Sie müssen zwei Buckets erstellen, einen Quell-Bucket und einen Ziel-Bucket. In den Beispielen wird zuerst die Bucket-ACL für den Ziel-Bucket aktualisiert. Anschließend werden der Protokollbereitstellungsgruppe die notwendigen Berechtigungen für das Schreiben von Protokollen zum Ziel-Bucket gewährt, bevor die Protokollierung für den Quell-Bucket aktiviert wird.

Diese Beispiele funktionieren nicht für Ziel-Buckets, die die Einstellung „Von Bucket-Besitzer erzwungen“ für Object Ownership verwenden.

Wenn der Ziel-Bucket die Einstellung „Von Besitzer erzwungen“ für Object Ownership verwendet, können Sie keine Bucket- oder Objekt-ACLs festlegen. Sie können auch keine Ziel-Erteilungen (Ziel-Erteilungen) in Ihre PutBucketLogging Konfiguration aufnehmen. Sie müssen eine Bucket-Richtlinie verwenden, um dem Protokollierungsserviceprinzipal (`logging.s3.amazonaws.com`) Zugriff zu gewähren. Weitere Informationen finden Sie unter Berechtigungen für Protokollbereitstellung.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

/// <summary>
/// This example shows how to enable logging on an Amazon Simple Storage
/// Service (Amazon S3) bucket. You need to have two Amazon S3 buckets for
/// this example. The first is the bucket for which you wish to enable
/// logging, and the second is the location where you want to store the
/// logs.
/// </summary>
public class ServerAccessLogging
{
    private static IConfiguration _configuration = null!;

    public static async Task Main()
    {
        LoadConfig();

        string bucketName = _configuration["BucketName"];
```

```
            string logBucketName = _configuration["LogBucketName"];
            string logObjectKeyPrefix = _configuration["LogObjectKeyPrefix"];
            string accountId = _configuration["AccountId"];

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the Amazon S3 client object's constructor.
            // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
            IAmazonS3 client = new AmazonS3Client();

            try
            {
                // Update bucket policy for target bucket to allow delivery of
logs to it.
                await SetBucketPolicyToAllowLogDelivery(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix,
                    accountId);

                // Enable logging on the source bucket.
                await EnableLoggingAsync(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }
        }

        /// <summary>
        /// This method grants appropriate permissions for logging to the
        /// Amazon S3 bucket where the logs will be stored.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
used
        /// to apply the bucket policy.</param>
        /// <param name="sourceBucketName">The name of the source bucket.</param>
        /// <param name="logBucketName">The name of the bucket where logging
        /// information will be stored.</param>
```

```
        /// <param name="logPrefix">The logging prefix where the logs should be
delivered.</param>
        /// <param name="accountId">The account id of the account where the
source bucket exists.</param>
        /// <returns>Async task.</returns>
        public static async Task SetBucketPolicyToAllowLogDelivery(
            IAmazonS3 client,
            string sourceBucketName,
            string logBucketName,
            string logPrefix,
            string accountId)
        {
            var resourceArn = @"""arn:aws:s3:::" + logBucketName + "/" +
logPrefix + @"*""";

            var newPolicy = @"{
                                ""Statement"":[{
                                ""Sid"": ""S3ServerAccessLogsPolicy"",
                                ""Effect"": ""Allow"",
                                ""Principal"": { ""Service"":
""logging.s3.amazonaws.com"" },
                                ""Action"": [""s3:PutObject""],
                                ""Resource"": [" + resourceArn + @"],
                                ""Condition"": {
                                ""ArnLike"": { ""aws:SourceArn"":
""arn:aws:s3:::" + sourceBucketName + @""" },
                                ""StringEquals"": { ""aws:SourceAccount"": """ +
accountId + @""" }
                                        }
                                    }]
                                }";
            Console.WriteLine($"The policy to apply to bucket {logBucketName} to
enable logging:");
            Console.WriteLine(newPolicy);

            PutBucketPolicyRequest putRequest = new PutBucketPolicyRequest
            {
                BucketName = logBucketName,
                Policy = newPolicy,
            };
            await client.PutBucketPolicyAsync(putRequest);
            Console.WriteLine("Policy applied.");
        }
```

```
        /// <summary>
        /// This method enables logging for an Amazon S3 bucket. Logs will be
 stored
        /// in the bucket you selected for logging. Selected prefix
        /// will be prepended to each log object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
 used
        /// to configure and apply logging to the selected Amazon S3 bucket.</
param>
        /// <param name="bucketName">The name of the Amazon S3 bucket for which
 you
        /// wish to enable logging.</param>
        /// <param name="logBucketName">The name of the Amazon S3 bucket where
 logging
        /// information will be stored.</param>
        /// <param name="logObjectKeyPrefix">The prefix to prepend to each
        /// object key.</param>
        /// <returns>Async task.</returns>
        public static async Task EnableLoggingAsync(
            IAmazonS3 client,
            string bucketName,
            string logBucketName,
            string logObjectKeyPrefix)
        {
            Console.WriteLine($"Enabling logging for bucket {bucketName}.");
            var loggingConfig = new S3BucketLoggingConfig
            {
                TargetBucketName = logBucketName,
                TargetPrefix = logObjectKeyPrefix,
            };

            var putBucketLoggingRequest = new PutBucketLoggingRequest
            {
                BucketName = bucketName,
                LoggingConfig = loggingConfig,
            };
            await client.PutBucketLoggingAsync(putBucketLoggingRequest);
            Console.WriteLine($"Logging enabled.");
        }

        /// <summary>
        /// Loads configuration from settings files.
        /// </summary>
```

```
        public static void LoadConfig()
        {
            _configuration = new ConfigurationBuilder()
                .SetBasePath(Directory.GetCurrentDirectory())
                .AddJsonFile("settings.json") // Load settings from .json file.
                .AddJsonFile("settings.local.json", true) // Optionally, load
  local settings.
                .Build();
        }
    }
```

- Weitere API-Informationen finden Sie unter PutBucketLogging in der APIAWS SDK for .NET -Referenz für .

Java

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketLoggingStatus;
import software.amazon.awssdk.services.s3.model.LoggingEnabled;
import software.amazon.awssdk.services.s3.model.PartitionedPrefix;
import software.amazon.awssdk.services.s3.model.PutBucketLoggingRequest;
import software.amazon.awssdk.services.s3.model.TargetObjectKeyFormat;

// Class to set a bucket policy on a target S3 bucket and enable server access
 logging on a source S3 bucket.
public class ServerAccessLogging {
    private static S3Client s3Client;

    public static void main(String[] args) {
        String sourceBucketName = "SOURCE-BUCKET";
        String targetBucketName = "TARGET-BUCKET";
        String sourceAccountId = "123456789012";
        String targetPrefix = "logs/";

        // Create S3 Client.
        s3Client = S3Client.builder().
                region(Region.US_EAST_2)
                .build();
```

```
        // Set a bucket policy on the target S3 bucket to enable server access
 logging by granting the
        // logging.s3.amazonaws.com principal permission to use the PutObject
 operation.
        ServerAccessLogging serverAccessLogging = new ServerAccessLogging();
        serverAccessLogging.setTargetBucketPolicy(sourceAccountId, sourceBucketName,
 targetBucketName);

        // Enable server access logging on the source S3 bucket.
        serverAccessLogging.enableServerAccessLogging(sourceBucketName,
 targetBucketName,
                targetPrefix);

    }

    // Function to set a bucket policy on the target S3 bucket to enable server
 access logging by granting the
    // logging.s3.amazonaws.com principal permission to use the PutObject operation.
    public void setTargetBucketPolicy(String sourceAccountId, String
 sourceBucketName, String targetBucketName) {
        String policy = "{\n" +
                "    \"Version\": \"2012-10-17\",\n" +
                "    \"Statement\": [\n" +
                "        {\n" +
                "            \"Sid\": \"S3ServerAccessLogsPolicy\",\n" +
                "            \"Effect\": \"Allow\",\n" +
                "            \"Principal\": {\"Service\": \"logging.s3.amazonaws.com
\"},\n" +
                "            \"Action\": [\n" +
                "                \"s3:PutObject\"\n" +
                "            ],\n" +
                "            \"Resource\": \"arn:aws:s3:::" + targetBucketName + "/*
\",\n" +
                "            \"Condition\": {\n" +
                "                \"ArnLike\": {\n" +
                "                    \"aws:SourceArn\": \"arn:aws:s3:::" +
 sourceBucketName + "\"\n" +
                "                },\n" +
                "                \"StringEquals\": {\n" +
                "                    \"aws:SourceAccount\": \"" + sourceAccountId +
 "\"\n" +
                "                }\n" +
                "            }\n" +
                "        }\n" +
```

```
                "    ]\n" +
                "}";
        s3Client.putBucketPolicy(b -> b.bucket(targetBucketName).policy(policy));
    }

    // Function to enable server access logging on the source S3 bucket.
    public void enableServerAccessLogging(String sourceBucketName, String
 targetBucketName,
            String targetPrefix) {
        TargetObjectKeyFormat targetObjectKeyFormat =
 TargetObjectKeyFormat.builder()

 .partitionedPrefix(PartitionedPrefix.builder().partitionDateSource("EventTime").build())
                .build();
        LoggingEnabled loggingEnabled = LoggingEnabled.builder()
                .targetBucket(targetBucketName)
                .targetPrefix(targetPrefix)
                .targetObjectKeyFormat(targetObjectKeyFormat)
                .build();
        BucketLoggingStatus bucketLoggingStatus = BucketLoggingStatus.builder()
                .loggingEnabled(loggingEnabled)
                .build();
        s3Client.putBucketLogging(PutBucketLoggingRequest.builder()
                .bucket(sourceBucketName)
                .bucketLoggingStatus(bucketLoggingStatus)
                .build());
    }

}
```

Verwenden der AWS CLI

Wir empfehlen Ihnen, in jedem , in dem Sie S3-Buckets haben AWS-Region , einen dedizierten Protokoll-Bucket zu erstellen. Anschließend können Sie Ihre Amazon-S3-Zugriffsprotokolle in diesem S3-Bucket bereitstellen lassen. Weitere Informationen und Beispiele finden Sie unter put-bucket-logging in der AWS CLI -Referenz.

Wenn der Ziel-Bucket die Einstellung „Von Besitzer erzwungen“ für Object Ownership verwendet, können Sie keine Bucket- oder Objekt-ACLs festlegen. Sie können auch keine Ziel-(Ziel)-Erteilungen in Ihre PutBucketLogging Konfiguration aufnehmen. Sie müssen eine Bucket-Richtlinie verwenden,

um dem Protokollierungsserviceprinzipal (`logging.s3.amazonaws.com`) Zugriff zu gewähren. Weitere Informationen finden Sie unter Berechtigungen für Protokollbereitstellung.

Example – Aktivieren von Zugriffsprotokollen für fünf Buckets in zwei Regionen

In diesem Beispiel werden die folgenden fünf Buckets verwendet:

- `1-DOC-EXAMPLE-BUCKET1-us-east-1`
- `2-DOC-EXAMPLE-BUCKET1-us-east-1`
- `3-DOC-EXAMPLE-BUCKET1-us-east-1`
- `1-DOC-EXAMPLE-BUCKET1-us-west-2`
- `2-DOC-EXAMPLE-BUCKET1-us-west-2`

> **Note**
>
> Der letzte Schritt im folgenden Verfahren enthält Bash-Beispielskripts, mit denen Sie Ihre Protokollierungs-Buckets erstellen und die Serverzugriffsprotokollierung für diese Buckets aktivieren können. Um diese Skripts verwenden zu können, müssen Sie die Dateien `policy.json` und `logging.json` erstellen wie im folgenden Verfahren beschrieben.

1. Erstellen Sie in den Regionen USA West (Oregon) und USA Ost (Nord-Virginia) zwei Ziel-Buckets für die Protokollierung mit den folgenden Namen:
   - `DOC-EXAMPLE-BUCKET1-logs-us-east-1`
   - `DOC-EXAMPLE-BUCKET1-logs-us-west-2`
2. Später im Verfahren aktivieren Sie die Serverzugriffsprotokollierung wie folgt:
   - `1-DOC-EXAMPLE-BUCKET1-us-east-1` protokolliert zum S3-Bucket `DOC-EXAMPLE-BUCKET1-logs-us-east-1` mit dem Präfix `1-DOC-EXAMPLE-BUCKET1-us-east-1`
   - `2-DOC-EXAMPLE-BUCKET1-us-east-1` protokolliert zum S3-Bucket `DOC-EXAMPLE-BUCKET1-logs-us-east-1` mit dem Präfix `2-DOC-EXAMPLE-BUCKET1-us-east-1`
   - `3-DOC-EXAMPLE-BUCKET1-us-east-1` protokolliert zum S3-Bucket `DOC-EXAMPLE-BUCKET1-logs-us-east-1` mit dem Präfix `3-DOC-EXAMPLE-BUCKET1-us-east-1`
   - `1-DOC-EXAMPLE-BUCKET1-us-west-2` protokolliert zum S3-Bucket `DOC-EXAMPLE-BUCKET1-logs-us-west-2` mit dem Präfix `1-DOC-EXAMPLE-BUCKET1-us-west-2`

- `2-DOC-EXAMPLE-BUCKET1-us-west-2` protokolliert zum S3-Bucket `DOC-EXAMPLE-BUCKET1-logs-us-west-2` mit dem Präfix `2-DOC-EXAMPLE-BUCKET1-us-west-2`

3. Gewähren Sie für jeden Ziel-Bucket über eine Bucket-ACL oder eine Bucket-Richtlinie Berechtigungen für die Serverzugriffsprotokoll-Bereitstellung:

   - Aktualisieren Sie die Bucket-Richtlinie (Empfohlen) – Verwenden Sie den folgenden `put-bucket-policy`-Befehl, um dem Prinzipal des Protokollierungsservices Berechtigungen zu erteilen. Ersetzen Sie *`DOC-EXAMPLE-DESTINATION-BUCKET-logs`* durch den Namen Ihres Ziel-Buckets.

     ```
     aws s3api put-bucket-policy --bucket DOC-EXAMPLE-DESTINATION-BUCKET-logs --policy
      file://policy.json
     ```

     `Policy.json` ist ein JSON-Dokument im aktuellen Ordner, das die folgende Bucket-Richtlinie enthält. Wenn Sie diese Bucket-Richtlinie verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre eigenen Informationen. In der folgenden Richtlinie ist *`DOC-EXAMPLE-DESTINATION-BUCKET-logs`* der Ziel-Bucket, in dem Serverzugriffsprotokolle bereitgestellt werden. *`DOC-EXAMPLE-SOURCE-BUCKET`* ist der Quell-Bucket. *`SOURCE-ACCOUNT-ID`* ist das AWS-Konto , das den Quell-Bucket besitzt.

     ```
     {
         "Version": "2012-10-17",
         "Statement": [
             {
                 "Sid": "S3ServerAccessLogsPolicy",
                 "Effect": "Allow",
                 "Principal": {
                     "Service": "logging.s3.amazonaws.com"
                 },
                 "Action": [
                     "s3:PutObject"
                 ],
                 "Resource": "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET-logs/*",
                 "Condition": {
                     "ArnLike": {
                         "aws:SourceArn": "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
                     },
                     "StringEquals": {
                         "aws:SourceAccount": "SOURCE-ACCOUNT-ID"
     ```

```
                }
            }
        }
    ]
}
```

- Aktualisieren der Bucket-ACL – Zum Erteilen von Berechtigungen für die S3-Protokollbereitstellungsgruppe verwenden Sie den folgenden `put-bucket-acl`-Befehl. Ersetzen Sie *`DOC-EXAMPLE-DESTINATION-BUCKET-logs`* durch den Namen Ihres Ziel-Buckets.

```
aws s3api put-bucket-acl --bucket DOC-EXAMPLE-DESTINATION-BUCKET-logs  --grant-
write URI=http://acs.amazonaws.com/groups/s3/LogDelivery --grant-read-acp
 URI=http://acs.amazonaws.com/groups/s3/LogDelivery
```

4. Erstellen Sie dann eine `logging.json`-Datei, die Ihre Protokollierungskonfiguration enthält (basierend auf einem der drei folgenden Beispiele). Nach der Erstellung der `logging.json`-Datei können Sie mit dem folgenden `put-bucket-logging`-Befehl die Protokollierungskonfiguration anwenden. Ersetzen Sie *`DOC-EXAMPLE-DESTINATION-BUCKET-logs`* durch den Namen Ihres Ziel-Buckets.

```
aws s3api put-bucket-logging --bucket DOC-EXAMPLE-DESTINATION-BUCKET-logs --bucket-
logging-status file://logging.json
```

> **Note**
>
> Anstatt mit diesem diesen `put-bucket-logging`-Befehl die Protokollierungskonfiguration auf jeden einzelnen Ziel-Bucket anzuwenden, können Sie eines der im nächsten Schritt bereitgestellten Bash-Skripts verwenden. Um diese Skripts verwenden zu können, müssen Sie die Dateien `policy.json` und `logging.json` erstellen wie in diesem Verfahren beschrieben.

Die `logging.json`-Datei ist ein JSON-Dokument im aktuellen Ordner, das die Protokollierungskonfiguration enthält. Wenn ein Ziel-Bucket die Einstellung „Von Bucket-Besitzer erzwungen“ für Object Ownership verwendet, kann Ihre Protokollierungskonfiguration keine Zielgewährungen enthalten. Weitere Informationen finden Sie unter Berechtigungen für Protokollbereitstellung.

Example – **`logging.json`** ohne Zielgewährungen

Die folgende `logging.json`-Beispieldatei enthält keine Zielgewährungen. Daher können Sie diese Konfiguration auf einen Ziel-Bucket anwenden, der die Einstellung „Von Bucket-Besitzer erzwungen“ für Objekt Ownership verwendet.

```
{
    "LoggingEnabled": {
        "TargetBucket": "DOC-EXAMPLE-DESTINATION-BUCKET-logs",
        "TargetPrefix": "DOC-EXAMPLE-DESTINATION-BUCKET/"
     }
 }
```

Example – **`logging.json`** mit Zielgewährungen

Die folgende `logging.json`-Beispieldatei enthält Zielgewährungen.

Wenn der Ziel-Bucket die Einstellung „Von Bucket-Besitzer erzwungen“ für Object Ownership verwendet, können Sie keine Zielgewährungen in Ihre PutBucketLogging-Konfiguration einfügen. Weitere Informationen finden Sie unter Berechtigungen für Protokollbereitstellung.

```
{
    "LoggingEnabled": {
        "TargetBucket": "DOC-EXAMPLE-DESTINATION-BUCKET-logs",
        "TargetPrefix": "DOC-EXAMPLE-DESTINATION-BUCKET/",
        "TargetGrants": [
             {
                "Grantee": {
                    "Type": "AmazonCustomerByEmail",
                    "EmailAddress": "user@example.com"
```

```
                },
                "Permission": "FULL_CONTROL"
            }
        ]
    }
}
```

Example – **`logging.json`** mit auf S3-Ereigniszeit festgelegtem Protokollobjekt-Schlüsselformat

In der folgenden `logging.json`-Datei wird das Protokollobjekt-Schlüsselformat in S3-Ereigniszeit geändert. Weitere Informationen zum Einrichten des Protokollobjekt-Schlüsselformats finden Sie unter the section called “Wie aktiviere ich die Protokollzustellung?”.

```
{
  "LoggingEnabled": {
      "TargetBucket": "DOC-EXAMPLE-DESTINATION-BUCKET-logs",
      "TargetPrefix": "DOC-EXAMPLE-DESTINATION-BUCKET/",
      "TargetObjectKeyFormat": {
          "PartitionedPrefix": {
              "PartitionDateSource": "EventTime"
          }
       }
  }
}
```

5. Sie können eines der folgenden Bash-Skripts verwenden, um allen Buckets in Ihrem Konto Zugriffsprotokollierung hinzuzufügen. Ersetzen Sie *`DOC-EXAMPLE-DESTINATION-BUCKET-logs`* durch den Namen Ihres Ziel-Buckets (Ziel-Buckets) und *`us-west-2`* durch den Namen der Region, in der sich Ihre Buckets befinden.

> **Note**
>
> Dieses Skript funktioniert nur, wenn sich alle Buckets in derselben Region befinden. Sind die Buckets auf mehrere Regionen verteilt, müssen Sie das Skript anpassen.

Example - Gewähren Sie Zugriff mit Bucket-Richtlinien und fügen Sie die Protokollierung für die Buckets in Ihrem Konto hinzu

```
  loggingBucket='DOC-EXAMPLE-DESTINATION-BUCKET-logs'
  region='us-west-2'


  # Create the logging bucket.
  aws s3 mb s3://$loggingBucket --region $region

  aws s3api put-bucket-policy --bucket $loggingBucket --policy file://policy.json

  # List the buckets in this account.
  buckets="$(aws s3 ls | awk '{print $3}')"

  # Put a bucket logging configuration on each bucket.
  for bucket in $buckets
      do
        # This if statement excludes the logging bucket.
        if [ "$bucket" != "$loggingBucket" ] ; then
            continue;
        fi
        printf '{
          "LoggingEnabled": {
            "TargetBucket": "%s",
            "TargetPrefix": "%s/"
        }
      }' "$loggingBucket" "$bucket"  > logging.json
      aws s3api put-bucket-logging --bucket $bucket --bucket-logging-status file://
logging.json
      echo "$bucket done"
  done

  rm logging.json

  echo "Complete"
```

Example - Gewähren Sie Zugriff mit Bucket-ACLs und fügen Sie die Protokollierung für die Buckets in Ihrem Konto hinzu

```
  loggingBucket='DOC-EXAMPLE-DESTINATION-BUCKET-logs'
  region='us-west-2'


  # Create the logging bucket.
  aws s3 mb s3://$loggingBucket --region $region

  aws s3api put-bucket-acl --bucket $loggingBucket --grant-write URI=http://
acs.amazonaws.com/groups/s3/LogDelivery --grant-read-acp URI=http://
acs.amazonaws.com/groups/s3/LogDelivery

  # List the buckets in this account.
  buckets="$(aws s3 ls | awk '{print $3}')"

  # Put a bucket logging configuration on each bucket.
  for bucket in $buckets
      do
        # This if statement excludes the logging bucket.
        if [ "$bucket" != "$loggingBucket" ] ; then
            continue;
        fi
        printf '{
          "LoggingEnabled": {
            "TargetBucket": "%s",
            "TargetPrefix": "%s/"
        }
      }' "$loggingBucket" "$bucket"  > logging.json
      aws s3api put-bucket-logging --bucket $bucket --bucket-logging-status file://
logging.json
      echo "$bucket done"
  done

  rm logging.json

  echo "Complete"
```

Überprüfen der Einrichtung Ihrer Serverzugriffsprotokolle

Führen Sie nach dem Aktivieren der Serverzugriffsprotokollierung die folgenden Schritte aus:

- Greifen Sie auf den Ziel-Bucket zu und überprüfen Sie, ob die Protokolldateien bereitgestellt werden. Nach dem Einrichten der Zugriffsprotokollierung kann es über eine Stunde dauern, bis alle Anforderungen ordnungsgemäß protokolliert und übermittelt wurden. Sie können die Protokollzustellung auch automatisch überprüfen, indem Sie Amazon S3-Anforderungsmetriken verwenden und Amazon- CloudWatch Alarme für diese Metriken einrichten. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.
- Vergewissern Sie sich, dass Sie den Inhalt der Protokolldateien öffnen und lesen können.

Informationen zur Fehlerbehebung bei der Serverzugriffsprotokollierung finden Sie unter Behebung von Fehlern bei der Server-Zugriffsprotokollierung.

Amazon-S3-Server-Zugriffsprotokollformat

Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anfragen, die an einen Amazon-S3-Bucket gestellt wurden. Sie können Serverzugriffsprotokolle für folgende Zwecke verwenden:

- Durchführung von Sicherheits- und Zugriffsprüfungen
- Kennenlernen Ihres Kundenstamms
- Nachvollziehen Ihrer Amazon-S3-Rechnung

Dieser Abschnitt beschreibt das Format und andere Details zu Amazon-S3-Serverzugriffprotokolldateien.

Die Server-Zugriffsprotokolldateien bestehen aus einer Reihe von durch Zeilenschaltungen voneinander getrennten Protokolldatensätzen. Jeder Protokolldatensatz stellt eine Anforderung dar und besteht aus durch Leerzeichen voneinander getrennter Felder.

Nachfolgend wird ein Beispielprotokoll mit sechs Protokolldatensätzen gezeigt.

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 3E57427F3EXAMPLE
 REST.GET.VERSIONING - "GET /DOC-EXAMPLE-BUCKET1?versioning HTTP/1.1" 200 - 113 - 7 -
```

```
 "-" "S3Console/0.4" - s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/
XV/VLi31234= SigV4 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader DOC-EXAMPLE-BUCKET1.s3.us-
west-1.amazonaws.com TLSV1.2 arn:aws:s3:us-west-1:123456789012:accesspoint/example-AP
 Yes
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 891CE47D2EXAMPLE
 REST.GET.LOGGING_STATUS - "GET /DOC-EXAMPLE-BUCKET1?logging HTTP/1.1" 200 -
 242 - 11 - "-" "S3Console/0.4" - 9vKBE6vMhrNiWHZmb2L0mXOcqPGzQOI5XLnCtZNPxev+Hf
+7tpT6sxDwDty4LHBUOZJG96N1234= SigV4 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader DOC-
EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2 - -
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be A1206F460EXAMPLE
 REST.GET.BUCKETPOLICY - "GET /DOC-EXAMPLE-BUCKET1?policy HTTP/1.1" 404
 NoSuchBucketPolicy 297 - 38 - "-" "S3Console/0.4" - BNaBsXZQQDbssi6xMBdBU2sLt
+Yf5kZDmeBUP35sFoKa3sLLeMC78iwEIWxs99CRUrbS4n11234= SigV4 ECDHE-RSA-AES128-GCM-SHA256
 AuthHeader DOC-EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2 - Yes
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:01:00 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 7B4A0FABBEXAMPLE
 REST.GET.VERSIONING - "GET /DOC-EXAMPLE-BUCKET1?versioning HTTP/1.1" 200 -
 113 - 33 - "-" "S3Console/0.4" - Ke1bUcazaN1jWuUlPJaxF64cQVpUEhoZKEG/hmy/gijN/
I1DeWqDfFvnpybfEseEME/u7ME1234= SigV4 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader DOC-
EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2 - -
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:01:57 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DD6CC733AEXAMPLE REST.PUT.OBJECT s3-dg.pdf "PUT /DOC-EXAMPLE-BUCKET1/
s3-dg.pdf HTTP/1.1" 200 - - 4406583 41754 28 "-" "S3Console/0.4" -
 10S62Zv81kBW7BB6SX4XJ48o6kpcl6LPwEoizZQQxJd5qDSCTLX0TgS37kYUBKQW3+bPdrg1234= SigV4
 ECDHE-RSA-AES128-SHA AuthHeader DOC-EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2
 - Yes
```

> **Note**
>
> Jedes Feld kann auf - gesetzt werden, um darauf hinzuweisen, dass die Daten unbekannt oder nicht verfügbar waren, oder dass das Feld für diese Anforderung nicht relevant war.

Themen

- Protokolldatensatzfelder

- Zusätzliche Protokollierung für KopierVorgänge
- Benutzerdefinierte Zugriffsprotokollinformationen
- Aspekte zur Programmierung des erweiterbaren Serverzugriff-Protokollformats

Protokolldatensatzfelder

In der folgenden Liste werden die wichtigsten Protokolldatensatzfelder beschrieben.

Bucket-Eigentümer

Die kanonische Benutzer-ID des Eigentümer des Quell-Buckets. Die kanonische Benutzer-ID ist eine weitere Form der AWS-Konto ID. Weitere Informationen zur kanonischen Benutzer-ID finden Sie unter AWS-Konto -Kennungen in der Allgemeine AWS-Referenz. Informationen darüber, wie Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Finding the canonical user ID for your AWS-Konto(Die kanonische Benutzer-ID für Ihr AWS-Konto finden).

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Bucket

Der Name des Buckets, für den die Anforderung verarbeitet wurde. Wenn das System eine fehlerhaft aufgebaute Anforderung erhält und den Bucket nicht bestimmen kann, erscheint die Anforderung nicht in einem Server-Zugriffsprotokoll.

Beispieleintrag

```
DOC-EXAMPLE-BUCKET1
```

Zeit

Die Uhrzeit, zu der die Anforderung empfangen wurde. Diese Datums- und Uhrzeitangaben entsprechen der Zeitzone UTC (Coordinated Universal Time). Das Format unter Verwendung der `strftime()`-Terminologie, nämlich: `[%d/%b/%Y:%H:%M:%S %z]`

Beispieleintrag

```
[06/Feb/2019:00:00:38 +0000]
```

Remote-IP

Die offensichtliche IP-Adresse des Anforderers. Auf dem Weg vorhandene Proxy-Server und Firewalls könnten die tatsächliche IP-Adresse des Computers verbergen, der die Anforderung gestellt hat.

Beispieleintrag

```
192.0.2.3
```

Auftraggeber

Die kanonische Benutzer-ID des Auftraggebers, oder - für nicht authentifizierte Anforderungen. Wenn der Anforderer ein IAM-Benutzer war, gibt dieses Feld den IAM-Benutzernamen des Anforderers zusammen mit dem zurück Root-Benutzer des AWS-Kontos , zu dem der IAM-Benutzer gehört. Diese ID ist dieselbe, die für den Zugriff zu Kontrollzwecken verwendet wird.

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Anforderungs-ID

Eine von Amazon S3 generierte Zeichenfolge, die jede Anforderung eindeutig identifiziert.

Beispieleintrag

```
3E57427F33A59F07
```

Operation

Die hier aufgeführte Operation ist deklariert al `SOAP.operation`, `REST.HTTP_method.resource_type`, `WEBSITE.HTTP_method.resource_type` oder `BATCH.DELETE.OBJECT`, oder `S3.action.resource_type` für Lebenszyklus und Protokollieren.

Beispieleintrag

```
REST.PUT.OBJECT
```

Schlüssel

Der Schlüsselteil (Objektname) der Anfrage.

Beispieleintrag

```
/photos/2019/08/puppy.jpg
```

Request-URI

Der `Request-URI`-Teil der HTTP-Anforderungsmeldung.

Beispieleintrag

```
"GET /DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg?x-foo=bar HTTP/1.1"
```

HTTP-Status

Der numerische HTTP-Statuscode der Antwort.

Beispieleintrag

```
200
```

Fehlercode

Der Amazon S3 Fehlercode oder `-`, wenn kein Fehler aufgetreten ist.

Beispieleintrag

```
NoSuchBucket
```

Gesendete Bytes

Die Anzahl der in der Antwort gesendeten Byte, ausgenommen HTTP-Protokoll-Overhead, oder `-`, falls null.

Beispieleintrag

```
2662992
```

Objektgröße

Die Gesamtgröße des betreffenden Objekts.

Beispieleintrag

```
3462992
```

Gesamtzeit

Die Anzahl der Millisekunden (ms), die die Anforderung aus Perspektive des Servers unterwegs war. Dieser Wert wird ab der Zeit gemessen, zu der Ihre Anforderung empfangen wurde, bis zu der Zeit, zu der das letzte Byte der Antwort gesendet wurde. Messungen aus der Perspektive des Clients dauern möglicherweise länger aufgrund der Netzwerklatenz.

Beispieleintrag

```
70
```

Umschlagzeit

Die Anzahl der Millisekunden, die Amazon S3 gebraucht hat, Ihre Anfrage zu verarbeiten. Dieser Wert wird ab der Zeit gemessen, zu der das letzte Byte Ihrer Anforderung empfangen wurde, bis zu der Zeit, zu der das erste Byte der Antwort gesendet wurde.

Beispieleintrag

```
10
```

Referer

Der Wert des HTTP-`Referer`-Headers, falls vorhanden. HTTP-Benutzeragenten (z. B. Browser) setzen diesen Header normalerweise auf die URL der verlinkenden oder einbettenden Seite, wenn eine Anforderung erfolgt.

Beispieleintrag

```
"http://www.example.com/webservices"
```

User-Agent

Der Wert des HTTP-`User-Agent`-Headers

Beispieleintrag

```
"curl/7.15.1"
```

Versions-ID

Die Versions-ID der Anforderung oder `-`, wenn die Operation keinen `versionId`-Parameter entgegennimmt.

Beispieleintrag

```
3HL4kqtJvjVBH40Nrjfkd
```

Host-ID

Die erweiterte Anforderungs-ID für `x-amz-id-2` oder Amazon S3

Beispieleintrag

```
s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/XV/VLi31234=
```

Signatur-Version

Die Signaturversion, `SigV2` oder `SigV4`, die für die Authentifizierung der Anforderung verwendet wurde, bzw. ein `-` für nicht authentifizierte Anforderungen.

Beispieleintrag

```
SigV2
```

Cipher Suite

Das Secure Sockets Layer (SSL)-Verschlüsselungsverfahren, das für die HTTPS-Anforderung ausgehandelt wurde bzw. ein `-` für HTTP.

Beispieleintrag

```
ECDHE-RSA-AES128-GCM-SHA256
```

Authentifizierungstyp

Die Art der verwendeten Anforderungsauthentifizierung: `AuthHeader` für Authentifizierungs-Header, `QueryString` für die Anforderungszeichenfolge (vorsignierte URL) oder ein `-` für nicht authentifizierte Anforderungen.

Beispieleintrag

```
AuthHeader
```

Host Header

Der für die Verbindung mit Amazon S3 verwendete Endpunkt.

Beispieleintrag

```
s3.us-west-2.amazonaws.com
```

Einige frühere Regionen unterstützen Legacy-Endpunkte. Möglicherweise sehen Sie diese Endpunkte in Ihren Serverzugriffsprotokollen oder - AWS CloudTrail protokollen. Weitere Informationen finden Sie unter Legacy-Endpunkte. Eine vollständige Liste der Amazon-S3-Regionen und -Endpunkte finden Sie unter Endpunkte und Kontingente von Amazon S3 in der Allgemeine Amazon Web Services-Referenz.

TLS-Version

Die vom Client ausgehandelte Transport Layer Security(TLS)-Version. Einer der folgenden Werte: `TLSv1.1`, `TLSv1.2`, `TLSv1.3` oder `-`, wenn TLS nicht verwendet wurde.

Beispieleintrag

```
TLSv1.2
```

Zugriffspunkt-ARN

Der Amazon-Ressourcenname (ARN) des Zugriffspunkts der Anforderung. Wenn der Zugriffspunkt-ARN fehlerhaft ist oder nicht verwendet wird, enthält das Feld ein `-`. Weitere Hinweise zu Zugangspunkten finden Sie unter Verwenden von Zugriffspunkten. Weitere

Informationen zu ARNs finden Sie unter Amazon-Ressourcenname (ARN) im AWS -Referenz-Handbuch.

Beispieleintrag

```
arn:aws:s3:us-east-1:123456789012:accesspoint/example-AP
```

aclRequired

Eine Zeichenfolge, die angibt, ob für die Anforderung eine Zugriffssteuerungsliste (ACL) für die Autorisierung erforderlich war. Wenn für die Anforderung eine ACL zur Autorisierung erforderlich war, lautet die Zeichenfolge `Yes`. Wenn keine ACLs erforderlich waren, lautet die Zeichenfolge `-`. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht. Weitere Hinweise zur Verwendung des Felds `aclRequired` zum Deaktivieren von ACLs finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Beispieleintrag

```
Yes
```

Zusätzliche Protokollierung für KopierVorgänge

Eine Kopieroperation umfasst ein `GET` und ein `PUT`. Aus diesem Grund zeichnen wir für eine Kopieroperation zwei Datensätze auf. Der vorherige Abschnitt beschreibt die Felder für den `PUT`-Teil der Operation. Die folgende Liste beschreibt die Felder in dem Datensatz, die sich auf den `GET`-Teil der Kopieroperation beziehen.

Bucket-Eigentümer

Die kanonische Benutzer-ID des Buckets, der das kopierte Objekt speichert. Die kanonische Benutzer-ID ist eine weitere Form der AWS-Konto ID. Weitere Informationen zur kanonischen Benutzer-ID finden Sie unter AWS-Konto -Kennungen in der Allgemeine AWS-Referenz. Informationen darüber, wie Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Finding the canonical user ID for your AWS-Konto(Die kanonische Benutzer-ID für Ihr AWS-Konto finden).

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Bucket

Der Name des Buckets, der das kopierte Objekt speichert.

Beispieleintrag

```
DOC-EXAMPLE-BUCKET1
```

Zeit

Die Uhrzeit, zu der die Anforderung empfangen wurde. Diese Datums- und Uhrzeitangaben entsprechen der Zeitzone UTC (Coordinated Universal Time). Das Format unter Verwendung der `strftime()`-Terminologie, nämlich: `[%d/%B/%Y:%H:%M:%S %z]`

Beispieleintrag

```
[06/Feb/2019:00:00:38 +0000]
```

Remote-IP

Die offensichtliche IP-Adresse des Anforderers. Auf dem Weg vorhandene Proxy-Server und Firewalls könnten die tatsächliche IP-Adresse des Computers verbergen, der die Anforderung gestellt hat.

Beispieleintrag

```
192.0.2.3
```

Auftraggeber

Die kanonische Benutzer-ID des Auftraggebers, oder `-` für nicht authentifizierte Anforderungen. Wenn der Anforderer ein IAM-Benutzer war, gibt dieses Feld den IAM-Benutzernamen des Anforderers zusammen mit dem zurück Root-Benutzer des AWS-Kontos , zu dem der IAM-Benutzer gehört. Diese ID ist dieselbe, die für den Zugriff zu Kontrollzwecken verwendet wird.

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Anforderungs-ID

Eine von Amazon S3 generierte Zeichenfolge, die jede Anforderung eindeutig identifiziert.

Beispieleintrag

```
3E57427F33A59F07
```

Operation

Die hier aufgeführte Operation ist deklariert al `SOAP.operation`, `REST.HTTP_method.resource_type`, `WEBSITE.HTTP_method.resource_type` oder `BATCH.DELETE.OBJECT`.

Beispieleintrag

```
REST.COPY.OBJECT_GET
```

Schlüssel

Der Schlüssel (Objektname) des kopierten Objekts oder `-`, wenn die Operation keinen Schlüsselparameter entgegennimmt.

Beispieleintrag

```
/photos/2019/08/puppy.jpg
```

Request-URI

Der `Request-URI`-Teil der HTTP-Anforderungsmeldung.

Beispieleintrag

```
"GET /DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg?x-foo=bar"
```

HTTP-Status

Der numerische HTTP-Statuscode des `GET`-Teils der Kopieroperation.

Beispieleintrag

```
200
```

Fehlercode

Der Fehlercode von Amazon S3 des GET-Teils des Kopiervorgangs oder -, wenn kein Fehler aufgetreten ist.

Beispieleintrag

```
NoSuchBucket
```

Gesendete Bytes

Die Anzahl der in der Antwort gesendeten Byte, ausgenommen HTTP-Protokoll-Overhead, oder -, falls null.

Beispieleintrag

```
2662992
```

Objektgröße

Die Gesamtgröße des betreffenden Objekts.

Beispieleintrag

```
3462992
```

Gesamtzeit

Die Anzahl der Millisekunden (ms), die die Anforderung aus Perspektive des Servers unterwegs war. Dieser Wert wird ab der Zeit gemessen, zu der Ihre Anforderung empfangen wurde, bis zu der Zeit, zu der das letzte Byte der Antwort gesendet wurde. Messungen aus der Perspektive des Clients dauern möglicherweise länger aufgrund der Netzwerklatenz.

Beispieleintrag

```
70
```

Umschlagzeit

Die Anzahl der Millisekunden, die Amazon S3 gebraucht hat, Ihre Anfrage zu verarbeiten. Dieser Wert wird ab der Zeit gemessen, zu der das letzte Byte Ihrer Anforderung empfangen wurde, bis zu der Zeit, zu der das erste Byte der Antwort gesendet wurde.

Beispieleintrag

```
10
```

Referer

Der Wert des HTTP-`Referer`-Headers, falls vorhanden. HTTP-Benutzeragenten (z. B. Browser) setzen diesen Header normalerweise auf die URL der verlinkenden oder einbettenden Seite, wenn eine Anforderung erfolgt.

Beispieleintrag

```
"http://www.example.com/webservices"
```

User-Agent

Der Wert des HTTP-`User-Agent`-Headers

Beispieleintrag

```
"curl/7.15.1"
```

Versions-ID

Die Version-ID des kopierten Objekts oder `-`, wenn der `x-amz-copy-source`-Header keinen `versionId`-Parameter als Teil der Kopierquelle angegeben hat.

Beispieleintrag

```
3HL4kqtJvjVBH40Nrjfkd
```

Host-ID

Die erweiterte Anforderungs-ID für `x-amz-id-2` oder Amazon S3

Beispieleintrag

```
s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/XV/VLi31234=
```

Signatur-Version

Die Signaturversion, `SigV2` oder `SigV4`, die für die Authentifizierung der Anforderung verwendet wurde, bzw. ein `-` für nicht authentifizierte Anforderungen.

Beispieleintrag

```
SigV4
```

Cipher Suite

Das Secure Sockets Layer (SSL)-Verschlüsselungsverfahren, das für die HTTPS-Anforderung ausgehandelt wurde bzw. ein `-` für HTTP.

Beispieleintrag

```
ECDHE-RSA-AES128-GCM-SHA256
```

Authentifizierungstyp

Die Art der verwendeten Anforderungsauthentifizierung: `AuthHeader` für Authentifizierungs-Header, `QueryString` für die Anforderungszeichenfolgen (vorsignierte URLs) oder ein `-` für nicht authentifizierte Anforderungen.

Beispieleintrag

```
AuthHeader
```

Host-Header

Der für die Verbindung mit Amazon S3 verwendete Endpunkt.

Beispieleintrag

```
s3.us-west-2.amazonaws.com
```

Einige frühere Regionen unterstützen Legacy-Endpunkte. Möglicherweise sehen Sie diese Endpunkte in Ihren Serverzugriffsprotokollen oder - AWS CloudTrail protokollen. Weitere Informationen finden Sie unter Legacy-Endpunkte. Eine vollständige Liste der Amazon-S3-Regionen und -Endpunkte finden Sie unter Endpunkte und Kontingente von Amazon S3 in der Allgemeine Amazon Web Services-Referenz.

TLS-Version

Die vom Client ausgehandelte Transport Layer Security(TLS)-Version. Einer der folgenden Werte: `TLSv1.1`, `TLSv1.2`, `TLSv1.3` oder `-`, wenn TLS nicht verwendet wurde.

Beispieleintrag

```
TLSv1.2
```

Zugriffspunkt-ARN

Der Amazon-Ressourcenname (ARN) des Zugriffspunkts der Anforderung. Wenn der Zugriffspunkt-ARN fehlerhaft ist oder nicht verwendet wird, enthält das Feld ein `-`. Weitere Hinweise zu Zugangspunkten finden Sie unter Verwenden von Zugriffspunkten. Weitere Informationen zu ARNs finden Sie unter Amazon-Ressourcenname (ARN) im AWS -Referenz-Handbuch.

Beispieleintrag

```
arn:aws:s3:us-east-1:123456789012:accesspoint/example-AP
```

aclRequired

Eine Zeichenfolge, die angibt, ob für die Anforderung eine Zugriffssteuerungsliste (ACL) für die Autorisierung erforderlich war. Wenn für die Anforderung eine ACL zur Autorisierung erforderlich war, lautet die Zeichenfolge `Yes`. Wenn keine ACLs erforderlich waren, lautet die Zeichenfolge `-`. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht. Weitere Hinweise zur Verwendung des Felds `aclRequired` zum Deaktivieren von ACLs finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Beispieleintrag

```
Yes
```

Benutzerdefinierte Zugriffsprotokollinformationen

Sie können benutzerdefinierte Informationen angeben, die im Zugriffsprotokolldatensatz für eine Anforderung gespeichert werden. Fügen Sie der URL für die Anforderung dazu einen benutzerdefinierten Abfragefolgenkettenparameter hinzu. Amazon S3 ignoriert Abfrage-Zeichenfolgenparameter, die mit `x-` beginnen, aber nimmt diese in den Zugriffsprotokoll-Datensatz für die Anforderung auf, als Teil des `Request-URI`-Felds des Protokolldatensatzes.

Beispielsweise verhält sich die Anfrage `GET` für `"s3.amazonaws.com/DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg?x-user=johndoe"` genauso wie die Anfrage für `"s3.amazonaws.com/DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg"`, abgesehen davon, dass die Zeichenfolge `"x-user=johndoe"` in das Feld `Request-URI` des entsprechenden Protokolldatensatzes eingefügt wird. Diese Funktionalität steht nur auf der REST-Schnittstelle zur Verfügung.

Aspekte zur Programmierung des erweiterbaren Serverzugriff-Protokollformats

Gelegentlich erweitern wir das Zugriffsprotokoll-Datensatzformat, indem wir am Ende jeder Zeile neue Felder hinzufügen. Daher sollten Sie sicherstellen, dass jeder Code, der Server-Zugriffsprotokolle analysiert, so geschrieben ist, dass er angefügte Felder verarbeiten kann, die er möglicherweise nicht versteht.

Löschen von Amazon-S3-Protokolldateien

Ein Amazon-S3-Bucket mit aktivierter Server-Zugriffsprotokollierung kann im Laufe der Zeit viele Server-Protokollobjekte ansammeln. Möglicherweise benötigt Ihre Anwendung diese Zugriffsprotokolle für einen bestimmten Zeitraum nachdem sie erstellt wurden und danach möchten Sie sie ggf. löschen. Sie können die Amazon-S3-Lebenszykluskonfiguration zum Erstellen von Regeln verwenden, sodass Amazon S3 diese Objekte am Ende ihres Lebenszyklus automatisch zum Löschen bereitstellt.

Sie können mithilfe eines gemeinsamen Präfixes für eine Teilmenge der Objekte in Ihrem S3-Bucket eine Lebenszykluskonfiguration definieren. Wenn Sie in Ihrer Konfiguration für die Server-Zugriffsprotokollierung ein Präfix angegeben haben, können Sie eine Lebenszyklus-Konfigurationsregel festlegen, um Protokollobjekte mit diesem Präfix zu löschen.

Angenommen, Ihre Protokollobjekte verwenden das Präfix `logs/`. Sie können dann über eine Lebenszyklus-Konfigurationsregel alle Objekte mit dem Präfix `logs/` im Bucket nach einem bestimmten Zeitraum löschen.

Weitere Informationen zur Lebenszyklus-Konfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Allgemeine Informationen zu Server-Zugriffsprotokollen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

Verwenden von Amazon-S3-Serverzugriffsprotokollen zur Identifizierung von Anforderungen

Sie können mittels Amazon-S3-Serverzugriffsprotokollen Amazon-S3-Anforderungen identifizieren.

> **Note**
>
> - Um Amazon S3-Anfragen zu identifizieren, empfehlen wir Ihnen, AWS CloudTrail Datenereignisse anstelle von Amazon S3-Serverzugriffsprotokollen zu verwenden. CloudTrail Datenereignisse sind einfacher einzurichten und enthalten mehr Informationen. Weitere Informationen finden Sie unter Identifizieren von Amazon S3-Anforderungen mit CloudTrail.
> - Je nachdem, wie viele Zugriffsanforderungen Sie erhalten, benötigt die Analyse Ihrer Protokolle möglicherweise mehr Ressourcen oder Zeit als die Verwendung von CloudTrail Datenereignissen.

Themen

Abfragen von Zugriffsprotokollen für Anforderungen über Amazon Athena

Sie können Amazon-S3-Anforderungen mit Amazon-S3-Zugriffsprotokollen mithilfe von Amazon Athena identifizieren.

Amazon S3 speichert Server-Zugriffsprotokolle als Objekte in einem S3-Bucket. Es ist oft einfacher, ein Tool zu verwenden, mit dem die Protokolle in Amazon S3 analysiert werden können. Athena unterstützt die Analyse von S3-Objekten und kann zur Abfrage von Amazon-S3-Zugriffsprotokollen verwendet werden.

Example

Das folgende Beispiel zeigt, wie Sie Amazon-S3-Server-Zugriffsprotokolle in Amazon Athena abfragen können. Ersetzen Sie die *user input placeholders* in den folgenden Beispielen durch eigene Daten.

> **Note**
>
> Zur Angabe eines Amazon-S3-Speicherorts in einer Athena-Abfrage müssen Sie einen S3-URI für den Ziel-Bucket bereitstellen, in dem Ihre Protokolle bereitgestellt werden. Dieser URI muss den Bucket-Namen und das Präfix im folgenden Format enthalten: `s3://DOC-EXAMPLE-BUCKET1-logs/prefix/`

1. Öffnen Sie die Athena-Konsole unter https://console.aws.amazon.com/athena/.
2. Führen Sie im Abfrage-Editor einen Befehl wie den folgenden aus. Ersetzen Sie *s3_access_logs_db* durch den Namen, den Sie Ihrer Datenbank geben möchten.

```
CREATE DATABASE s3_access_logs_db
```

> **Note**
>
> Es hat sich bewährt, die Datenbank in derselben AWS-Region wie Ihren S3-Bucket zu erstellen.

3. Führen Sie im Abfrage-Editor einen Befehl wie den folgenden aus, um in der in Schritt 2 erstellten Datenbank ein Tabellenschema zu erstellen. Ersetzen Sie *s3_access_logs_db.mybucket_logs* durch den Namen, den Sie Ihrer Tabelle geben möchten. Die Datentypwerte `STRING` und `BIGINT` sind die Zugriffsprotokolleigenschaften. Sie können diese Eigenschaften in Athena abfragen. Geben Sie für `LOCATION` wie oben erwähnt den Pfad von S3-Bucket und Präfix ein.

```
CREATE EXTERNAL TABLE `s3_access_logs_db.mybucket_logs`(
  `bucketowner` STRING,
  `bucket_name` STRING,
  `requestdatetime` STRING,
  `remoteip` STRING,
  `requester` STRING,
```

```
  `requestid` STRING,
  `operation` STRING,
  `key` STRING,
  `request_uri` STRING,
  `httpstatus` STRING,
  `errorcode` STRING,
  `bytessent` BIGINT,
  `objectsize` BIGINT,
  `totaltime` STRING,
  `turnaroundtime` STRING,
  `referrer` STRING,
  `useragent` STRING,
  `versionid` STRING,
  `hostid` STRING,
  `sigv` STRING,
  `ciphersuite` STRING,
  `authtype` STRING,
  `endpoint` STRING,
  `tlsversion` STRING,
  `accesspointarn` STRING,
  `aclrequired` STRING)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  'input.regex'='([^ ]*) ([^ ]*) \\[(.*?)\\] ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*) (\"[^\"]*\"|-) (-|[0-9]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 (\"[^\"]*\"|-) ([^ ]*)(?: ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*))?.*$')
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET1-logs/prefix/'
```

4. Wählen Sie im Navigationsbereich unter Database (Datenbank) die Datenbank aus.
5. Wählen Sie unter Tables (Tabellen)neben dem Namen der Tabelle Preview table (Tabellenvorschau) aus.

   Im Fensterbereich Results (Ergebnisse) sollten Daten aus den Server-Zugriffsprotokollen angezeigt werden, also `bucketowner`, `bucket`, `requestdatetime` usw. Dies bedeutet, dass die Athena-Tabelle erfolgreich erstellt wurde. Sie können jetzt die Amazon-S3-Server-Zugriffsprotokolle abfragen.

Example – Anzeigen, wer ein Objekt um welche Uhrzeit (Zeitstempel, IP-Adresse und IAM-Benutzer) gelöscht hat

```
SELECT requestdatetime, remoteip, requester, key
FROM s3_access_logs_db.mybucket_logs
WHERE key = 'images/picture.jpg' AND operation like '%DELETE%';
```

Example – Anzeigen aller Vorgänge, die von einem IAM-Benutzer ausgeführt wurden

```
SELECT *
FROM s3_access_logs_db.mybucket_logs
WHERE requester='arn:aws:iam::123456789123:user/user_name';
```

Example – Anzeigen aller Vorgänge, die in einem bestimmten Zeitraum für ein Objekt ausgeführt wurden

```
SELECT *
FROM s3_access_logs_db.mybucket_logs
WHERE Key='prefix/images/picture.jpg'
    AND parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
    BETWEEN parse_datetime('2017-02-18:07:00:00','yyyy-MM-dd:HH:mm:ss')
    AND parse_datetime('2017-02-18:08:00:00','yyyy-MM-dd:HH:mm:ss');
```

Example – Anzeigen der Menge der in einem festgelegten Zeitraum an eine bestimmten IP-Adresse übertragenen Daten

```
SELECT coalesce(SUM(bytessent), 0) AS bytessenttotal
FROM s3_access_logs_db.mybucket_logs
WHERE remoteip='192.0.2.1'
AND parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2022-06-01','yyyy-MM-dd')
AND parse_datetime('2022-07-01','yyyy-MM-dd');
```

> **Note**
>
> Zur Reduzierung der Beibehaltungsdauer Ihrer Protokolle können Sie für Ihren Serverzugriffsprotokoll-Bucket eine S3-Lebenszykluskonfiguration erstellen. Erstellen Sie Lebenszykluskonfigurationsregeln, um Protokolldateien regelmäßig zu entfernen. Dadurch wird die Menge der Daten reduziert, die Athena in einer Abfrage analysiert. Weitere Informationen finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

Identifizieren von Signature-Version-2-Anforderungen mittels Amazon-S3-Zugriffsprotokollen

Die Amazon-S3-Unterstützung für Signature Version 2 wird deaktiviert (veraltet). Danach akzeptiert Amazon S3 keine Anforderungen mit Signature Version 2 mehr, alle Anforderungen müssen also mit Signature Version 4 signiert werden. Sie können Signature-Version-2-Zugriffsanforderungen mittels Amazon-S3-Zugriffsprotokollen identifizieren.

> **Note**
>
> Um Signature Version 2-Anforderungen zu identifizieren, empfehlen wir, dass Sie AWS CloudTrail Datenereignisse anstelle von Amazon S3-Serverzugriffsprotokollen verwenden. CloudTrail Datenereignisse sind einfacher einzurichten und enthalten mehr Informationen als Serverzugriffsprotokolle. Weitere Informationen finden Sie unter Identifizieren von Anforderungen von Amazon S3 Signature Version 2 mithilfe von CloudTrail.

Example – Anzeigen aller Anforderer, die Signature Version 2-Datenverkehr senden

```
SELECT requester, sigv, Count(sigv) as sigcount
FROM s3_access_logs_db.mybucket_logs
GROUP BY requester, sigv;
```

Identifizieren von Objektzugriffsanforderungen mittels Amazon-S3-Zugriffsprotokollen

Sie können Abfragen an Amazon-S3-Server-Zugriffsprotokolle verwenden, um Amazon-S3-Objektzugriffsanforderungen für Vorgänge zu identifizieren, wie etwa `GET`, `PUT` und `DELETE`, und weitere Informationen über diese Anforderungen zu entdecken.

Das folgende Amazon-Athena-Abfragebeispiel zeigt, wie Sie alle PUT-Objektanforderungen für Amazon S3 aus einem Serverzugriffsprotokoll abrufen können.

Example – Anzeigen aller Anforderer, die **PUT**-Objektanforderungen in einem bestimmten Zeitraum senden

```
SELECT bucket_name, requester, remoteip, key, httpstatus, errorcode, requestdatetime
FROM s3_access_logs_db
WHERE operation='REST.PUT.OBJECT' AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Das folgende Amazon Athena-Abfragebeispiel zeigt, wie alle GET-Objektanfragen für Amazon S3 aus dem Server-Zugriffsprotokoll abgerufen werden.

Example – Anzeigen aller Anforderer, die **GET**-Objektanforderungen in einem bestimmten Zeitraum senden

```
SELECT bucket_name, requester, remoteip, key, httpstatus, errorcode, requestdatetime
FROM s3_access_logs_db
WHERE operation='REST.GET.OBJECT' AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Das folgende Amazon Athena-Abfragebeispiel zeigt, wie alle anonymen Anforderungen aus dem Server-Zugriffsprotokoll in Ihre S3-Buckets gelangen.

Example – Anzeigen aller anonymen Anforderer, die in einem bestimmten Zeitraum Anforderungen an einen Bucket richten

```
SELECT bucket_name, requester, remoteip, key, httpstatus, errorcode, requestdatetime
FROM s3_access_logs_db.mybucket_logs
WHERE requester IS NULL AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
```

```
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Die folgende Amazon-Athena-Abfrage zeigt, wie alle Anforderungen für Ihre S3-Buckets identifiziert werden, für die eine Zugriffssteuerungsliste (ACL) zur Autorisierung erforderlich war. Sie können diese Informationen verwenden, um diese ACL-Berechtigungen zu den entsprechenden Bucket-Richtlinien zu migrieren und ACLs zu deaktivieren. Nachdem Sie diese Bucket-Richtlinien erstellt haben, können Sie ACLs für diese Buckets deaktivieren. Weitere Informationen über das Deaktivieren von ACLs finden Sie unter Voraussetzungen für die Deaktivierung von ACLs.

Example – Identifizieren Sie alle Anforderungen, für die eine ACL zur Autorisierung erforderlich war

```
SELECT bucket_name, requester, key, operation, aclrequired, requestdatetime
FROM s3_access_logs_db
WHERE aclrequired = 'Yes' AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2022-05-10:00:00:00','yyyy-MM-dd:HH:mm:ss')
AND parse_datetime('2022-08-10:00:00:00','yyyy-MM-dd:HH:mm:ss')
```

Note

- Sie können den Datumsbereich nach Belieben an Ihre Anforderungen anpassen.
- Diese Abfragebeispiele können auch für die Sicherheitsüberwachung nützlich sein. Sie können die Ergebnisse auf `PutObject`- oder `GetObject`-Aufrufe von unerwarteten oder nicht autorisierten IP-Adressen oder Anforderern und zum Identifizieren anonymer Anforderungen an Ihre Buckets prüfen.
- Diese Abfrage ruft nur Informationen von der Zeit ab, zu der die Protokollierung aktiviert wurde.
- Wenn Sie - AWS CloudTrail Protokolle verwenden, finden Sie weitere Informationen unter Identifizieren des Zugriffs auf S3-Objekte mithilfe von CloudTrail.

Überwachen von Metriken mit Amazon CloudWatch

Amazon- CloudWatch Metriken für Amazon S3 können Ihnen helfen, die Leistung von Anwendungen, die Amazon S3 verwenden, zu verstehen und zu verbessern. Es gibt mehrere Möglichkeiten, CloudWatch mit Amazon S3 zu verwenden.

Tägliche Speichermetriken für Buckets

Überwachen Sie den Bucket-Speicher mit CloudWatch, das Speicherdaten von Amazon S3 sammelt und zu lesbaren täglichen Metriken verarbeitet. Diese Speichermetriken für Amazon S3 werden einmal pro Tag gemeldet und allen Kunden ohne zusätzliche Kosten zur Verfügung gestellt.

Anforderungsmetriken

Überwachen von Amazon-S3-Anforderungen, um Probleme bei der Ausführung schnell zu identifizieren und zu beheben. Die Metriken stehen in 1-Minuten-Intervallen nach einer gewissen Latenz für die Verarbeitung zur Verfügung. Diese CloudWatch Metriken werden zum gleichen Tarif wie die CloudWatch benutzerdefinierten Amazon-Metriken abgerechnet. Weitere Informationen zu CloudWatch Preisen finden Sie unter Amazon- CloudWatch Preise. Weitere Informationen zum optionalen Erhalt dieser Metriken finden Sie unter CloudWatch - Metrikkonfigurationen.

Wenn die Anforderungsmetriken aktiviert sind, werden sie für alle Objekt-Vorgänge gemeldet. Standardmäßig stehen diese 1-Minuten-Metriken auf Ebene der Amazon-S3-Buckets zur Verfügung. Sie können auch einen Filter für die Metriken definieren, die mit einem gemeinsamen Präfix oder Objekt-Tag oder Zugriffspunkt erfasst wurden.

- Zugriffspunkt – Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind und vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in S3. Mit dem Zugriffspunkt-Filter erhalten Sie Erkenntnisse in Ihre Zugriffspunkt-Nutzung. Weitere Hinweise zu Zugangspunkten finden Sie unter Überwachen und Protokollieren von Zugriffspunkten.
- Präfix – Obwohl das Amazon-S3-Datenmodell eine flache Struktur aufweist, können Sie die Hierarchie durch die Verwendung eines Präfixes inferieren. Ein Präfix ähnelt einem Verzeichnisnamen, mit dem Sie ähnliche Objekte in einem Bucket gruppieren können. Die S3-Konsole unterstützt Präfixe mit dem Ordnerkonzept. Wenn Sie nach dem Präfix filtern, werden Objekte mit demselben Präfix in die Metrik-Konfiguration aufgenommen. Weitere Informationen zu Präfixen finden Sie unter Organisieren von Objekten mit Präfixen.
- Tag – Tags sind Schlüssel-Wert-Namenspaare, die Sie Objekten hinzufügen können. Mit Markierungen können Sie Objekte einfacher finden und organisieren. Tags können auch als Filter für Metrikkonfigurationen verwendet werden, sodass nur Objekte mit diesen Tags in die Metrikkonfiguration aufgenommen werden. Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

Um diese Metriken an bestimmten Geschäftsanwendungen, Workflows oder internen Organisationen auszurichten, können Sie nach einem freigegebenen Präfix, einem Objekt-Tag oder einem Zugriffspunkt filtern.

Replikationsmetriken

Replikationsmetriken – Überwachen der Gesamtzahl der S3-API-Operationen mit ausstehender Replikation, der Gesamtgröße der Objekte mit ausstehender Replikation, der maximalen Replikationszeit in der Ziel- AWS-Region und der Gesamtzahl der Operationen, deren Replikation fehlgeschlagen ist. Replikationsregeln, bei denen S3-Replikationszeitkontrolle (S3 RTC) oder S3-Replikationsmetriken aktiviert sind, veröffentlichen Replikationsmetriken.

Weitere Informationen finden Sie unter Überwachen des Fortschritts mit Replikationsmetriken und S3-Ereignisbenachrichtigungen oder Erfüllen der Compliance-Anforderungen mit S3-Replikationszeitkontrolle (S3 RTC).

Metriken von Amazon S3 Storage Lens

Sie können Nutzungs- und Aktivitätsmetriken von S3 Storage Lens in Amazon veröffentlichen, CloudWatch um eine einheitliche Ansicht Ihres Betriebszustands in CloudWatch Dashboards zu erstellen. S3-Storage-Lens-Metriken sind im `AWS/S3/Storage-Lens`-Namespace verfügbar. Die CloudWatch Veröffentlichungsoption ist für S3-Storage-Lens-Dashboards verfügbar, die auf erweiterte Metriken und Empfehlungen aktualisiert wurden. Sie können die CloudWatch Veröffentlichungsoption für eine neue oder vorhandene Dashboard-Konfiguration in S3 Storage Lens aktivieren.

Weitere Informationen finden Sie unter Überwachen von Metriken von S3 Storage Lens in CloudWatch.

Alle CloudWatch Statistiken werden für einen Zeitraum von 15 Monaten aufbewahrt, damit Sie auf historische Informationen zugreifen und einen besseren Überblick über die Leistung Ihrer Webanwendung oder Ihres Services erhalten. Weitere Informationen zu CloudWatchfinden Sie unter Was ist Amazon CloudWatch? im Amazon- CloudWatch Benutzerhandbuch. Abhängig von Ihren Anwendungsfällen benötigen Sie möglicherweise einige zusätzliche Konfigurationen für Ihre CloudWatch Alarme. Sie können beispielsweise einen mathematischen Metrik-Ausdruck verwenden, um einen Alarm zu erzeugen. Weitere Informationen finden Sie unter Verwenden von CloudWatch Metriken, Verwenden von Metrikberechnungen, Verwenden von Amazon- CloudWatch Alarmen und Erstellen eines CloudWatch Alarms basierend auf einem metrischen mathematischen Ausdruck im Amazon- CloudWatch Benutzerhandbuch.

Bereitstellung von Best-Effort- CloudWatch Metriken

CloudWatch -Metriken werden auf Best-Effort-Basis bereitgestellt. Die meisten Anfragen für ein Amazon S3-Objekt mit Anforderungsmetriken führen dazu, dass ein Datenpunkt an gesendet wird CloudWatch.

Die Vollständigkeit und Rechtzeitigkeit der Metriken ist nicht garantiert. Der Datenpunkt für eine bestimmte Anforderung wird möglicherweise mit einem Zeitstempel zurückgegeben, der nach der tatsächlichen Anforderungsverarbeitung liegt. Der Datenpunkt kann sich um eine Minute verzögern, bevor er über verfügbar ist CloudWatch, oder er wird möglicherweise überhaupt nicht bereitgestellt. CloudWatch request-Metriken geben Ihnen eine Vorstellung von der Art des Datenverkehrs zu Ihrem Bucket in nahezu Echtzeit. Sie sind nicht als vollständige Abrechnung aller Anforderungen vorgesehen.

Aufgrund der Best-Effort-Natur dieser Funktion enthalten die im Fakturierungs- und Kostenverwaltungs-Dashboard verfügbaren Berichte möglicherweise eine oder mehrere Zugriffsanforderungen, die nicht in den Bucket-Metriken angezeigt werden.

Weitere Informationen finden Sie unter den folgenden Themen.

Themen

- Metriken und Dimensionen
- Zugreifen auf CloudWatch Metriken
- CloudWatch -Metrikkonfigurationen

Metriken und Dimensionen

Die Speichermetriken und Dimensionen, die Amazon S3 an Amazon sendet, CloudWatch sind in den folgenden Tabellen aufgeführt.

Bereitstellung von Best-Effort- CloudWatch Metriken

CloudWatch -Metriken werden auf Best-Effort-Basis bereitgestellt. Die meisten Anfragen für ein Amazon S3-Objekt mit Anforderungsmetriken führen dazu, dass ein Datenpunkt an gesendet wird CloudWatch.

Die Vollständigkeit und Rechtzeitigkeit der Metriken ist nicht garantiert. Der Datenpunkt für eine bestimmte Anforderung wird möglicherweise mit einem Zeitstempel zurückgegeben, der nach der tatsächlichen Anforderungsverarbeitung liegt. Der Datenpunkt kann sich um eine Minute verzögern,

bevor er über verfügbar ist CloudWatch, oder er wird möglicherweise überhaupt nicht bereitgestellt. CloudWatch request-Metriken geben Ihnen eine Vorstellung von der Art des Datenverkehrs zu Ihrem Bucket in nahezu Echtzeit. Sie sind nicht als vollständige Abrechnung aller Anforderungen vorgesehen.

Aufgrund der Best-Effort-Natur dieser Funktion enthalten die im Fakturierungs- und Kostenverwaltungs-Dashboard verfügbaren Berichte möglicherweise eine oder mehrere Zugriffsanforderungen, die nicht in den Bucket-Metriken angezeigt werden.

Themen

Tägliche Amazon S3-Speichermetriken für Buckets in CloudWatch

Der `AWS/S3`-Namespace enthält die folgenden täglichen Speichermetriken für Buckets.

Metrik	Beschreibung
`BucketSizeBytes`	Die Datenmenge in Byte, die in einem Bucket in den folgenden Speicherklassen gespeichert wird: • S3 Standard (`STANDARD`) • S3 Intelligent-Tiering (`INTELLIGENT_TIERING`) • S3 Standard-Infrequent Access (`STANDARD_IA`) • S3 One Zone – seltener Zugriff (`ONEZONE_IA`) • Reduced Redundancy Storage (RRS) (`REDUCED_REDUNDANCY`)

Metrik	Beschreibung
	• S3 Glacier Instant Retrieval (`GLACIER_IR`) • S3 Glacier Deep Archive (`DEEP_ARCHIVE`) • S3 Glacier Flexible Retrieval (`GLACIER`) • S3 Express One Zone (`EXPRESS_ONEZONE`) Zur Berechnung dieses Werts wird die Größe aller (aktuellen und nicht aktuellen) Objekte und Metadaten im Bucket summiert – einschließlich der Größe aller Teile für sämtliche unvollständige mehrteilige Uploads in den Bucket. Gültige Speichertypfilter: `StandardStorage`, `IntelligentTieringFAStorage`, `IntelligentTieringIAStorage`, `IntelligentTieringAAStorage`, `IntelligentTieringAIAStorage`, `IntelligentTieringDAAStorage`, `StandardIAStorage`, `StandardIASizeOverhead`, `StandardIAObjectOverhead`, `OneZoneIAStorage`, `OneZoneIASizeOverhead`, `ReducedRedundancyStorage`, `GlacierInstantRetrievalSizeOverhead` `GlacierInstantRetrievalStorage`, `GlacierStorage`, `GlacierStagingStorage`, `GlacierObjectOverhead`, `GlacierS3ObjectOverhead`, `DeepArchiveStorage`, `DeepArchiveObjectOverhead`, `DeepArchiveS3ObjectOverhead`, `DeepArchiveStagingStorage` und `ExpressOneZone` (siehe Dimension `StorageType`) Einheiten: Byte Gültige Statistiken: Durchschnitt

Metrik	Beschreibung
`NumberOfObjects`	Die Gesamtzahl der Objekte für alle Speicherklassen, die in einem Bucket für allgemeine Zwecke gespeichert sind. Zur Berechnung dieses Werts werden alle aktuellen und nicht aktuellen Objekte im Bucket, alle Löschmarkierungen sowie die Gesamtanzahl der Teile sämtliche r unvollständiger mehrteiliger Uploads in den Bucket gezählt. Bei Verzeichnis-Buckets mit Objekten in der Speicherklasse S3 Express One Zone wird dieser Wert berechnet, indem alle Objekte im Bucket gezählt werden, enthält jedoch nicht unvollständige Mehrfach-Uploads in den Bucket. Gültige Speichertypfilter: nur `AllStorageTypes` (siehe `StorageType` -Dimension) Einheiten: Anzahl Gültige Statistiken: Durchschnitt

Amazon S3-Anforderungsmetriken in CloudWatch

Der `AWS/S3`-Namespace enthält die folgenden Anforderungsmetriken. Zu diesen Metriken gehören nicht fakturierbare Anfragen (im Falle von `GET` Anfragen von `CopyObject` und Replikation).

> **Note**
>
> Amazon S3-Anforderungsmetriken in CloudWatch werden für Verzeichnis-Buckets nicht unterstützt.

Metrik	Beschreibung
`AllRequests`	Die Gesamtanzahl von HTTP-Anfragen an einen Amazon-S3-Bucket, unabhängig vom Typ. Wenn Sie eine Metrikkonfiguration mit einem Filter verwenden, gibt diese Metrik nur die HTTP-Anfragen zurück, die den Filteranforderungen entsprechen. Einheiten: Anzahl

Metrik	Beschreibung
	Gültige Statistiken: Summe
`GetRequests`	Die Gesamtanzahl von HTTP `GET`-Anforderungen an Objekte in einem Amazon-S3-Bucket. Das Auflisten von Vorgängen ist hierin nicht inbegriff en. Diese Metrik wird für die Quelle jeder `CopyObject` Anforderung erhöht. Einheiten: Anzahl Gültige Statistiken: Summe **Note** Paginierte listenorientierte Anforderungen, wie ListMultipartUploa ds, ListPartsListObjectVersions, und andere, sind in dieser Metrik nicht enthalten.
`PutRequests`	Die Gesamtanzahl von HTTP `PUT`-Anforderungen an Objekte in einem Amazon-S3-Bucket. Diese Metrik wird für das Ziel jeder `CopyObject` Anforderung erhöht. Einheiten: Anzahl Gültige Statistiken: Summe
`DeleteRequests`	Die Gesamtanzahl von HTTP `DELETE`-Anforderungen an Objekte in einem Amazon-S3-Bucket. Diese Metrik enthält auch -DeleteObj ectsAnforderungen. Diese Metrik zeigt die Anzahl der gestellten Anfragen, nicht die Anzahl der gelöschten Objekte an. Einheiten: Anzahl Gültige Statistiken: Summe

Metrik	Beschreibung
`HeadRequests`	Die Gesamtanzahl von HTTP `HEAD`-Anfragen an einen Amazon-S3-Bucket. Einheiten: Anzahl Gültige Statistiken: Summe
`PostRequests`	Die Gesamtanzahl von HTTP `POST`-Anfragen an einen Amazon-S3-Bucket. Einheiten: Anzahl Gültige Statistiken: Summe **Note** DeleteObjects - und -SelectObjectContentAnforderungen sind nicht in dieser Metrik enthalten.
`SelectRequests`	Die Anzahl der Amazon S3-SelectObjectContentAnforderungen für Objekte in einem Amazon S3-Bucket. Einheiten: Anzahl Gültige Statistiken: Summe
`SelectBytesScanned`	Die Anzahl der Bytes der Daten, die mit Amazon S3-SelectObjectContentAnforderungen in einem Amazon S3-Bucket gescannt wurden. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9

Metrik	Beschreibung
`SelectBytesReturned`	Die Anzahl der Bytes der Daten, die mit Amazon S3-SelectObjectContentAnforderungen in einem Amazon S3-Bucket zurückgegeben wurden. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`ListRequests`	Die Gesamtanzahl von HTTP-Anforderungen zum Auflisten der Inhalte eines Buckets. Einheiten: Anzahl Gültige Statistiken: Summe
`BytesDownloaded`	Anzahl der heruntergeladenen Bytes für Anfragen an einen Amazon-S3-Bucket, wobei die Antwort einen Textkörper enthält. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`BytesUploaded`	Anzahl der hochgeladenen Bytes für Anfragen an einen Amazon-S3-Bucket, wobei die Antwort einen Textkörper enthält. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9

Metrik	Beschreibung
`4xxErrors`	Die Anzahl der HTTP-4xx-Client-Fehlerstatuscodeanforderungen an einen Amazon S3-Bucket mit einem Wert von 0 oder 1. Die Durchschnittsstatistik zeigt die Fehlerrate und die Summenstatistik die Anzahl dieses Fehlertyps für den jeweiligen Zeitraum an. Einheiten: Anzahl Gültige Statistiken: Durchschnitt (Berichte pro Anforderung), Summe (Berichte pro Zeitraum), Min, Max, Stichprobenanzahl
`5xxErrors`	Die Anzahl der HTTP 5xx-Server-Fehlerstatuscodeanforderungen an einen Amazon S3-Bucket mit einem Wert von 0 oder 1. Die Durchschnittsstatistik zeigt die Fehlerrate und die Summenstatistik die Anzahl dieses Fehlertyps für den jeweiligen Zeitraum an. Einheiten: Anzahl Gültige Statistiken: Durchschnitt (Berichte pro Anforderung), Summe (Berichte pro Zeitraum), Min, Max, Stichprobenanzahl
`FirstByteLatency`	Zeit pro Anforderung vom Zeitpunkt des Eingangs der vollständigen Anfragen bei einem Amazon-S3-Bucket bis das Zurückgeben der Antwort beginnt. Einheiten: Millisekunden Gültige Statistiken: Durchschnitt, Summe, Min, Max (entspricht p100), Stichprobenanzahl, beliebiges Perzentil zwischen p0,0 und p100

Metrik	Beschreibung
`TotalRequestLatency`	Zeit pro Anforderung vom Zeitpunkt des Eingangs des ersten Bytes bei einem Amazon-S3-Bucket bis zum Senden des letzten Bytes. Diese Metrik umfasst die erforderliche Zeit, um den Textkörper zu erhalten und den Antworttextkörper zu senden, welche nicht in `FirstByteLatency` enthalten ist. Einheiten: Millisekunden Gültige Statistiken: Durchschnitt, Summe, Min, Max (entspricht p100), Stichprobenanzahl, beliebiges Perzentil zwischen p0,0 und p100

S3-Replikationsmetriken in CloudWatch

Sie können den Fortschritt der Replikation mit S3-Replikationsmetriken überwachen, indem Sie ausstehende Bytes, ausstehende Operationen und die Replikationslatenz nachverfolgen. Weitere Informationen finden Sie unter Überwachen des Fortschritts mit Replikationsmetriken.

> **Note**
>
> Sie können Alarme für Ihre Replikationsmetriken in Amazon aktivieren CloudWatch. Wenn Sie Alarme für Ihre Replikationsmetriken einrichten, stellen Sie das Feld Missing data treatment (Behandlung fehlender Daten) auf Treat missing data as ignore (maintain the alarm state) (Fehlende Daten ignorieren) (Alarmstatus beibehalten) ein.

Metrik	Beschreibung
`ReplicationLatency`	Die maximale Anzahl von Sekunden, um die sich das Replikationsziel hinter der Quelle AWS-Region für eine bestimmte Replikationsregel AWS-Region befindet. Einheiten: Sekunden Gültige Statistiken: Max

Metrik	Beschreibung
`BytesPendingReplication`	Die Gesamtanzahl der Bytes von Objekten, die für eine bestimmte Replikationsregel ausstehen. Einheiten: Byte Gültige Statistiken: Max
`OperationsPendingReplication`	Die Anzahl der Vorgänge mit ausstehender Replikation für eine bestimmte Replikationsregel. Einheiten: Anzahl Gültige Statistiken: Max
`OperationsFailedReplication`	Die Anzahl der Operationen, deren Replikation für eine bestimmte Replikationsregel fehlgeschlagen ist. Einheiten: Anzahl Gültige Statistiken: Summe (Gesamtzahl der fehlgeschlagenen Operationen), Durchschnitt (Fehlerrate), Stichprobenanzahl (Gesamtzahl der Replikationsoperationen)

S3-Storage-Lens-Metriken in CloudWatch

Sie können Nutzungs- und Aktivitätsmetriken von S3 Storage Lens in Amazon veröffentlichen, CloudWatch um eine einheitliche Ansicht Ihres Betriebszustands in CloudWatch Dashboards zu erstellen. S3-Storage-Lens-Metriken werden im `AWS/S3/Storage-Lens` Namespace in veröffentlicht CloudWatch. Die CloudWatch Veröffentlichungsoption ist für S3-Storage-Lens-Dashboards verfügbar, die auf erweiterte Metriken und Empfehlungen aktualisiert wurden.

Eine Liste der Metriken von S3 Storage Lens, die in veröffentlicht werden CloudWatch, finden Sie unter Amazon S3-Storage-Lens-Metrikglossar. Eine vollständige Liste der Dimensionen finden Sie unter Dimensionen.

Anforderungsmetriken für S3 Object Lambda in CloudWatch

S3 Objekt Lambda enthält die folgenden Anforderungsmetriken.

Metrik	Beschreibung
`AllRequests`	Die Gesamtanzahl von HTTP-Anforderungen an einen Amazon-S3-Bucket mithilfe eines Object Lambda Access Point. Einheiten: Anzahl Gültige Statistiken: Summe
`GetRequests`	Die Gesamtanzahl von HTTP-`GET`-Anforderungen für Objekte mithilfe eines Object Lambda Access Point. Diese Metrik umfasst keine Operationen zum Auflisten. Einheiten: Anzahl Gültige Statistiken: Summe
`BytesUploaded`	Die Anzahl von Bytes, die mithilfe eines Object Lambda Access Point in einen Amazon-S3-Bucket hochgeladen wurden, wobei die Anforderung einen Textkörper enthält. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`PostRequests`	Die Gesamtanzahl von HTTP-`POST`-Anfragen an einen Amazon-S3-Bucket mithilfe eines Object Lambda Access Point. Einheiten: Anzahl Gültige Statistiken: Summe
`PutRequests`	Die Gesamtanzahl von HTTP-`PUT`-Anfragen für Objekte in einem Amazon-S3-Bucket mithilfe eines Object Lambda Access Point. Einheiten: Anzahl Gültige Statistiken: Summe

Metrik	Beschreibung
`DeleteRequests`	Die Gesamtanzahl von HTTP-`DELETE`-Anfragen für Objekte in einem Amazon-S3-Bucket mithilfe eines Object Lambda Access Point. Diese Metrik umfasst -DeleteObjectsAnforderungen. Diese Metrik zeigt die Anzahl der gestellten Anfragen, nicht die Anzahl der gelöschten Objekte an. Einheiten: Anzahl Gültige Statistiken: Summe
`BytesDownloaded`	Die Anzahl der heruntergeladenen Bytes für Anforderungen an einen Amazon-S3-Bucket mithilfe eines Object Lambda Access Point, wobei die Antwort einen Textkörper enthält Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`FirstByte Latency`	Zeit pro Anforderung vom Zeitpunkt des Eingangs der vollständigen Anfragen bei einem Amazon-S3-Bucket über einen Object Lambda Access Point bis zum Beginn des Zurückgebens der Antwort. Diese Metrik hängt von der Laufzeit der AWS Lambda -Funktion zum Transformieren des Objekts ab, bevor die Funktion die Bytes an den Object Lambda Access Point zurückgibt. Einheiten: Millisekunden Gültige Statistiken: Durchschnitt, Summe, Min, Max (entspricht p100), Stichprobenanzahl, beliebiges Perzentil zwischen p0,0 und p100

Metrik	Beschreibung
`TotalRequestLatency`	Zeit pro Anforderung vom Zeitpunkt des Eingangs des ersten Bytes bis zum Senden des letzten Bytes an einen Object Lambda Access Point. Diese Metrik umfasst die erforderliche Zeit, um den Textkörper zu erhalten und den Antworttextkörper zu senden, welche nicht in `FirstByteLatency` enthalten ist. Einheiten: Millisekunden Gültige Statistiken: Durchschnitt, Summe, Min, Max (entspricht p100), Stichprobenanzahl, beliebiges Perzentil zwischen p0,0 und p100
`HeadRequests`	Die Gesamtanzahl von HTTP-`HEAD`-Anfragen an einen Amazon-S3-Bucket mithilfe eines Object Lambda Access Point. Einheiten: Anzahl Gültige Statistiken: Summe
`ListRequests`	Die Gesamtanzahl von HTTP-`GET`-Anforderungen zum Auflisten der Inhalte eines Amazon-S3-Buckets. Diese Metrik enthält sowohl `ListObjects` - als auch `ListObjectsV2` -Operationen. Einheiten: Anzahl Gültige Statistiken: Summe
`4xxErrors`	Die Anzahl der HTTP-4xx-Server-Fehlerstatuscodeanforderungen an einen Amazon S3-Bucket unter Verwendung eines Object Lambda Access Point mit einem Wert von 0 oder 1. Die Durchschnittsstatistik zeigt die Fehlerrate und die Summenstatistik die Anzahl dieses Fehlertyps für den jeweiligen Zeitraum an. Einheiten: Anzahl Gültige Statistiken: Durchschnitt (Berichte pro Anforderung), Summe (Berichte pro Zeitraum), Min, Max, Stichprobenanzahl

Metrik	Beschreibung
`5xxErrors`	Die Anzahl der HTTP 5xx-Server-Fehlerstatuscodeanforderungen an einen Amazon S3-Bucket unter Verwendung eines Object Lambda Access Point mit einem Wert von 0 oder 1. Die Durchschnittsstatistik zeigt die Fehlerrate und die Summenstatistik die Anzahl dieses Fehlertyps für den jeweiligen Zeitraum an. Einheiten: Anzahl Gültige Statistiken: Durchschnitt (Berichte pro Anforderung), Summe (Berichte pro Zeitraum), Min, Max, Stichprobenanzahl
`ProxiedRequests`	Die Anzahl der HTTP-Anforderungen an einen Object Lambda Access Point, die die standardmäßige Amazon-S3-API-Antwort zurückgeben. (Für solche Anforderungen ist keine Lambda-Funktion konfiguriert.) Einheiten: Anzahl Gültige Statistiken: Summe
`InvokedLambda`	Die Anzahl der HTTP-Anforderungen an ein S3-Objekt, bei denen eine Lambda-Funktion aufgerufen wurde. Einheiten: Anzahl Gültige Statistiken: Summe
`LambdaResponseRequests`	Die Anzahl von `WriteGetObjectResponse` -Anforderungen, die von der Lambda-Funktion gestellt wurden. Diese Metrik gilt nur für `GetObject` -Anforderungen.
`LambdaResponse4xx`	Die Anzahl der HTTP-4xx-Clientfehler, die beim Aufrufen `WriteGetObjectResponse` von von einer Lambda-Funktion auftreten. Diese Metrik liefert dieselben Informationen wie `4xxErrors` , jedoch nur für `WriteGetObjectResponse` -Aufrufe.

Metrik	Beschreibung
`LambdaResponse5xx`	Die Anzahl der HTTP-5xx-Serverfehler, die beim Aufrufen `WriteGetObjectResponse` von von einer Lambda-Funktion auftreten. Diese Metrik liefert dieselben Informationen wie `5xxErrors` , jedoch nur für `WriteGetObjectResponse` -Aufrufe.

Amazon S3-on-Outposts-Metriken in CloudWatch

Eine Liste der Metriken in CloudWatch , die für S3-on-Outposts-Buckets verwendet werden, finden Sie unter CloudWatch-Metriken.

Amazon S3-Dimensionen in CloudWatch

Die nachstehenden Dimensionen werden verwendet, um Amazon-S3-Metriken zu filtern.

Dimension	Beschreibung
`BucketName`	Diese Dimension filtert die angeforderten Daten nur für den identifizierten Bucket.
`StorageType`	Diese Dimension filtert die Daten, die Sie in einem Bucket gespeichert haben, nach den folgenden Speichertypen: • `StandardStorage` – Die Gesamtzahl der Bytes für Objekte in der `STANDARD`-Speicherklasse. • `IntelligentTieringAAStorage` – Die Anzahl von Bytes für Objekte in Archive Access Tier der `INTELLIGENT_TIERING` -Speicherklasse. • `IntelligentTieringAIAStorage` – Die Anzahl von Bytes für Objekte in Stufe Archive Instant Access Tier der `INTELLIGENT_TIERING` -Speicherklasse. • `IntelligentTieringDAAStorage` – Die Anzahl von Bytes für Objekte in Deep Archive Access Tier der `INTELLIGENT_TIERING` -Speicherklasse.

Dimension	Beschreibung
	• `IntelligentTieringFAStorage` – Die Anzahl von Bytes für Objekte in Frequent Access Tier der `INTELLIGENT_TIERING` -Speicherklasse. • `IntelligentTieringIAStorage` – Die Anzahl von Bytes für Objekte in Infrequent Access Tier der `INTELLIGENT_TIERING` -Speicherklasse. • `StandardIAStorage` – Die Anzahl der Bytes für Objekte in der `STANDARD_IA` Speicherklasse S3 Standard-Infrequent Access (). • `StandardIASizeOverhead` – Die Anzahl von Bytes für Objekte mit einer Größe von weniger als 128 KB in der `STANDARD_IA` -Speicherklasse. • `IntAAObjectOverhead` – Für jedes Objekt in der `INTELLIGENT_TIERING` -Speicherklasse in Archive Access Tier fügt S3 Glacier 32 KB Speicher für Index und zugehörige Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederher zustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen die Gebühren für S3 Glacier Flexible Retrieval in Rechnung gestellt. • `IntAAS3ObjectOverhead` – Für jedes Objekt in der `INTELLIGENT_TIERING` -Speicherklasse in Archive Access Tier verwendet Amazon S3 8 KB Speicher für den Namen des Objekts und andere Metadaten. Für diesen zusätzlichen Speicherplatz werden Ihnen -S3-Standardgebühren in Rechnung gestellt. • `IntDAAObjectOverhead` – Für jedes Objekt in der `INTELLIGENT_TIERING` -Speicherklasse in Deep Archive Access Tier fügt S3 Glacier 32 KB Speicher für Index und zugehörige Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederher zustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen

Dimension	Beschreibung
	die S3 Glacier Deep Archive-Speichergebühren in Rechnung gestellt. • `IntDAAS3ObjectOverhead` – Für jedes Objekt in der `INTELLIGENT_TIERING` -Speicherklasse in Deep Archive Access Tier fügt Amazon S3 8 KB Speicher für Index und zugehörige Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederher zustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen -S3-Standardgebühren in Rechnung gestellt. • `OneZoneIAStorage` – Die Anzahl von Bytes für Objekte in der Speicherklasse S3 One Zone-Infrequent Access (`ONEZONE_IA`). • `OneZoneIASizeOverhead` – Die Anzahl von Bytes für Objekte mit einer Größe von weniger als 128 KB in der `ONEZONE_IA` -Speicherklasse. • `ReducedRedundancyStorage` – Die Anzahl von Bytes für Objekte in der RRS-Speicherklasse (Reduced Redundancy Storage). • `GlacierInstantRetrievalSizeOverhead` – Die Anzahl von Bytes für Objekte, die kleiner als 128 KB sind, in der Speicherklasse S3 Glacier Instant Retrieval. • `GlacierInstantRetrievalStorage` – Die Anzahl von Bytes für Objekte in der Speicherklasse S3 Glacier Instant Retrieval. • `GlacierStorage` – Die Anzahl von Bytes für Objekte in der Speicherklasse S3 Glacier Flexible Retrieval. • `GlacierStagingStorage` – Die Anzahl von Bytes für Teile von mehrteiligen Objekten, bevor die `CompleteMultipartUpload` -Anforderung für Objekte in der Speicherklasse S3 Glacier Flexible Retrieval abgeschlossen ist. • `GlacierObjectOverhead` – Für jedes archivierte Objekt fügt S3 Glacier 32 KB Speicher für den Index und zugehörig

Dimension	Beschreibung
	e Metadaten hinzu. Diese zusätzlichen Daten sind erforderl ich, um Ihr Objekt zu identifizieren und wiederherzustellen . Für diesen zusätzlichen Speicherplatz werden Ihnen die Gebühren für S3 Glacier Flexible Retrieval in Rechnung gestellt. • `GlacierS3ObjectOverhead` – Für jedes in S3 Glacier Flexible Retrieval archivierte Objekt verwendet Amazon S3 8 KB Speicher für den Namen des Objekts und andere Metadaten. Für diesen zusätzlichen Speicherplatz werden Ihnen -S3-Standardgebühren in Rechnung gestellt. • `DeepArchiveStorage` – Die Anzahl von Bytes für Objekte in der Speicherklasse S3 Glacier Deep Archive. • `DeepArchiveObjectOverhead` – Für jedes archivier te Objekt fügt S3 Glacier 32 KB Speicher für den Index und zugehörige Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederher zustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen die S3 Glacier Deep Archive-Gebühren in Rechnung gestellt. • `DeepArchiveS3ObjectOverhead` – Für jedes in S3 Glacier Deep Archive archivierte Objekt verwendet Amazon S3 8 KB Speicher für den Namen des Objekts und andere Metadaten. Für diesen zusätzlichen Speicherplatz werden Ihnen -S3-Standardgebühren in Rechnung gestellt. • `DeepArchiveStagingStorage` – Die Anzahl von Bytes für Teile von mehrteiligen Objekten, bevor die `CompleteM ultipartUpload` -Anforderung für Objekte in der Speicherklasse S3 Glacier Deep Archive abgeschlossen ist. • `ExpressOneZone` – Die Gesamtzahl der Bytes für Objekte in der S3-Express-One-Zone-Speicherklasse.

Dimension	Beschreibung
`FilterId`	Diese Dimension filtert Metrikkonfigurationen, die Sie für Anforderungsmetriken für einen Bucket angeben. Wenn Sie eine Metrikkonfiguration erstellen, geben Sie eine Filter-ID an (z. B. ein Präfix, ein Tag oder einen Zugriffspunkt). Weitere Informati onen finden Sie unter Erstellen einer Metrik-Konfiguration.

S3-Replikationsdimensionen in CloudWatch

Die folgenden Dimensionen werden verwendet, um S3-Replikationsmetriken zu filtern.

Dimension	Beschreibung
`SourceBucket`	Der Name der Bucket-Objekte wird repliziert.
`DestinationBucket`	Der Name der Bucket-Objekte wird in repliziert.
`RuleId`	Eine eindeutige Kennung für die Regel, die diese Replikati onsmetrik zur Aktualisierung ausgelöst hat.

Dimensionen von S3 Storage Lens in CloudWatch

Eine Liste der Dimensionen, die zum Filtern von S3-Storage-Lens-Metriken in verwendet werden CloudWatch, finden Sie unter Dimensionen.

S3-Objekt-Lambda-Anforderungsdimensionen in CloudWatch

Die folgenden Dimensionen werden verwendet, um Daten aus einem Object Lambda Access Point zu filtern.

Dimension	Beschreibung
`AccessPointName`	Der Name des Zugriffspunkts, von dem Anforderungen gestellt werden.
`DataSourceARN`	Die Quelle, aus der der Object Lambda Access Point die Daten abruft. Wenn die Anforderung eine Lambda-Funktion aufruft, bezieht sich dies

Dimension	Beschreibung
	auf den Amazon-Ressourcennamen (ARN) von Lambda. Ansonsten bezieht sich dies auf den ARN des Zugriffspunkts.

Zugreifen auf CloudWatch Metriken

Sie können die folgenden Vorgehensweisen nutzen, um die Speichermetriken von Amazon S3 anzuzeigen. Damit Amazon-S3-Metriken berücksichtigt werden, müssen Sie Zeitstempel für Anfang und Ende angeben. Für Metriken für einen bestimmten 24-Stunden-Zeitraum setzen Sie das Zeitintervall auf 86400 Sekunden, die Anzahl der Sekunden eines Tages. Denken Sie daran, die Dimensionen `BucketName` und `StorageType` festzulegen.

Verwenden der AWS CLI

Wenn Sie beispielsweise die verwenden möchten, AWS CLI um den Durchschnitt der Größe eines bestimmten Buckets in Bytes zu erhalten, können Sie den folgenden Befehl verwenden:

```
aws cloudwatch get-metric-statistics --metric-name BucketSizeBytes --namespace AWS/S3
 --start-time 2016-10-19T00:00:00Z --end-time 2016-10-20T00:00:00Z --statistics Average
 --unit Bytes --region us-west-2 --dimensions Name=BucketName,Value=DOC-EXAMPLE-BUCKET
 Name=StorageType,Value=StandardStorage --period 86400 --output json
```

Dieses Beispiel erzeugt die folgende Ausgabe.

```
{
    "Datapoints": [
        {
            "Timestamp": "2016-10-19T00:00:00Z",
            "Average": 1025328.0,
            "Unit": "Bytes"
        }
    ],
    "Label": "BucketSizeBytes"
}
```

Verwenden der S3-Konsole

So zeigen Sie Metriken mithilfe der Amazon- CloudWatch Konsole an

1. Öffnen Sie die - CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Metrics (Metriken) aus.
3. Wählen Sie den Namespace S3 aus.
4. (Optional) Um eine Metrik anzuzeigen, geben Sie den Metriknamen in das Suchfeld ein.
5. (Optional) Um nach der StorageType Dimension zu filtern, geben Sie den Namen der Speicherklasse in das Suchfeld ein.

So zeigen Sie eine Liste gültiger Metriken an, die für Ihr gespeichert sind AWS-Konto , indem Sie die verwenden AWS CLI

- Geben Sie als Eingabeaufforderung den folgenden Befehl ein.

```
aws cloudwatch list-metrics --namespace "AWS/S3"
```

Weitere Informationen zu den Berechtigungen, die für den Zugriff auf CloudWatch Dashboards erforderlich sind, finden Sie unter Amazon- CloudWatch Dashboard-Berechtigungen im Amazon-CloudWatch Benutzerhandbuch.

CloudWatch -Metrikkonfigurationen

Mit Amazon- CloudWatch Anforderungsmetriken für Amazon S3 können Sie einminütige CloudWatch Metriken erhalten, CloudWatch Alarme festlegen und auf CloudWatch Dashboards zugreifen, um den near-real-time Betrieb und die Leistung Ihres Amazon S3-Speichers anzuzeigen. Für Anwendungen, die von Cloud-Speicher abhängig sind, ermöglichen diese Metriken Ihnen, Betriebsprobleme schnell zu identifizieren und entsprechende Maßnahmen zu ergreifen. Wenn diese 1-Minuten-Metriken aktiviert sind, stehen sie standardmäßig auf Ebene der Amazon-S3-Buckets zur Verfügung.

Wenn Sie die CloudWatch Anforderungsmetriken für die Objekte in einem Bucket abrufen möchten, müssen Sie eine Metrikkonfiguration für den Bucket erstellen. Weitere Informationen finden Sie unter Erstellen einer CloudWatch Metrikkonfiguration für alle Objekte in Ihrem Bucket.

Sie können auch ein freigegebenes Präfix, Objekt-Tags oder einen Zugriffspunkt verwenden, um einen Filter für die gesammelten Metriken zu definieren. Mit dieser Methode zum Definieren

eines Filters können Sie also Metrikfilter an bestimmte Geschäftsanwendungen, Workflows oder interne Organisationen anpassen. Weitere Informationen finden Sie unter Erstellen einer Metrik-Konfiguration, die nach dem Präfix, Objekt-Tag oder Zugriffspunkt filtert. Weitere Informationen zu den verfügbaren CloudWatch-Metriken und den Unterschieden zwischen Speicher- und Anforderungsmetriken finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

Berücksichtigen Sie bei Verwendung von Metrik-Konfigurationen Folgendes:

- Sie können maximal 1 000 Metrik-Konfigurationen pro Bucket verwenden.
- Unter Verwendung von Filtern können Sie wählen, welche Objekte in einem Bucket Metrik-Konfigurationen enthalten sollen. Sie können nach einem freigegebenen Präfix, Objekt-Tag oder Zugriffspunkt filtern, um Metrikfilter an bestimmte Geschäftsanwendungen, Workflows oder interne Organisationen auszurichten. Um Metriken für den gesamten Bucket anzufordern, erstellen Sie eine Metrik-Konfiguration ohne Filter.
- Metrik-Konfigurationen sind nur erforderlich, um Anforderungsmetriken zu aktivieren. Tägliche Speichermetriken auf Bucket-Ebene sind immer aktiviert und werden allen Kunden ohne zusätzliche Kosten zur Verfügung gestellt. Derzeit ist es nicht möglich, tägliche Speichermetriken für eine gefilterte Untermenge von Objekten zu erhalten.
- Jede Metrik-Konfiguration unterstützt den gesamten Satz verfügbarer Anforderungsmetriken. Operationsspezifische Metriken (z. B. `PostRequests`) werden nur gemeldet, wenn es Anforderungen dieses Typs für den Bucket oder Filter gibt.
- Anforderungsmetriken werden für Vorgänge auf Objektebene gemeldet. Sie werden außerdem für Vorgänge gemeldet, die Bucket-Inhalte auflisten, beispielsweise GET Bucket (List Objects), GET Bucket Object Versions und List Multipart Uploads, nicht jedoch für andere Operationen an Buckets.
- Anforderungsmetriken unterstützen das Filtern mit Präfixen, Objekt-Tags oder Zugriffspunkten, Speichermetriken dagegen nicht.

Bereitstellung von Best-Effort- CloudWatch Metriken

CloudWatch -Metriken werden auf Best-Effort-Basis bereitgestellt. Die meisten Anfragen für ein Amazon S3-Objekt mit Anforderungsmetriken führen dazu, dass ein Datenpunkt an gesendet wird CloudWatch.

Die Vollständigkeit und Rechtzeitigkeit der Metriken ist nicht garantiert. Der Datenpunkt für eine bestimmte Anforderung wird möglicherweise mit einem Zeitstempel zurückgegeben, der nach der

tatsächlichen Anforderungsverarbeitung liegt. Der Datenpunkt kann sich um eine Minute verzögern, bevor er über verfügbar ist CloudWatch, oder er wird möglicherweise überhaupt nicht bereitgestellt. CloudWatch request-Metriken geben Ihnen eine Vorstellung von der Art des Datenverkehrs zu Ihrem Bucket in nahezu Echtzeit. Sie sind nicht als vollständige Abrechnung aller Anforderungen vorgesehen.

Aufgrund der Best-Effort-Natur dieser Funktion enthalten die im Fakturierungs- und Kostenverwaltungs-Dashboard verfügbaren Berichte möglicherweise eine oder mehrere Zugriffsanforderungen, die nicht in den Bucket-Metriken angezeigt werden.

Weitere Informationen zum Arbeiten mit CloudWatch Metriken in Amazon S3 finden Sie in den folgenden Themen.

Themen

- Erstellen einer CloudWatch Metrikkonfiguration für alle Objekte in Ihrem Bucket
- Erstellen einer Metrik-Konfiguration, die nach dem Präfix, Objekt-Tag oder Zugriffspunkt filtert
- Löschen eines Metrikfilters

Erstellen einer CloudWatch Metrikkonfiguration für alle Objekte in Ihrem Bucket

Wenn Sie Anforderungsmetriken konfigurieren, können Sie eine CloudWatch Metrikkonfiguration für alle Objekte in Ihrem Bucket erstellen oder nach Präfix, Objekt-Tag oder Zugriffspunkt filtern. Die Verfahren in diesem Thema zeigen Ihnen, wie Sie eine Konfiguration für alle Objekte in Ihrem Bucket erstellen. Informationen zum Erstellen einer Konfiguration, die nach Objekt-Tag, Präfix oder Zugriffspunkt filtert, finden Sie unter Erstellen einer Metrik-Konfiguration, die nach dem Präfix, Objekt-Tag oder Zugriffspunkt filtert.

Es gibt drei Arten von Amazon- CloudWatch Metriken für Amazon S3: Speichermetriken, Anforderungsmetriken und Replikationsmetriken. Speichermetriken werden einmal pro Tag gemeldet und allen Kunden ohne zusätzliche Kosten zur Verfügung gestellt. Die Anforderungsmetriken sind in 1-Minuten-Intervallen nach einer gewissen Latenz für die Verarbeitung verfügbar. Anforderungsmetriken werden zum CloudWatch Standardtarif abgerechnet. Sie wählen Anforderungsmetriken aus, indem Sie diese in der Konsole konfigurieren oder die Amazon-S3-API verwenden. S3-Replikationsmetriken enthalten detaillierte Metriken für die Replikationsregeln in Ihrer Replikationskonfiguration. Mit Replikationsmetriken können Sie den minute-by-minute Fortschritt überwachen, indem Sie ausstehende Bytes, ausstehende Operationen, Operationen, bei denen die Replikation fehlgeschlagen ist, und die Replikationslatenz verfolgen.

Weitere Informationen zu CloudWatch Metriken für Amazon S3 finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

Sie können einem Bucket über die Amazon-S3-Konsole, mit der AWS Command Line Interface (AWS CLI) oder Amazon-S3-REST-API Metrik-Konfigurationen hinzufügen.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets (Buckets) den Namen des Buckets mit den Objekte aus, für die Sie Anforderungsmetriken anfordern möchten.
3. Wählen Sie den Tab Metrics (Metriken).
4. Wählen Sie unter Bucket metrics (Bucket-Metriken) View additional charts (Zusätzliche Diagramme anzeigen) aus.
5. Wählen Sie den Tab Request metrics (Anforderungsmetriken).
6. Wählen Sie Create Filter (Filter erstellen).
7. Geben Sie im Feld Filter name (Filtername) Ihren Filternamen ein.

   Namen dürfen nur Buchstaben, Zahlen, Punkte, Bindestriche und Unterstriche enthalten. Wir empfehlen, den Namen `EntireBucket` für einen Filter zu verwenden, der für alle Objekte gilt.
8. Wählen Sie unter Filter scope (Filterbereich) die Option This filter applies to all objects in the bucket (Dieser Filter gilt für alle Objekte im Bucket).

   Sie können auch einen Filter definieren, sodass die Metriken nur für eine Untermenge von Objekten im Bucket erfasst und gemeldet werden. Weitere Informationen finden Sie unter Erstellen einer Metrik-Konfiguration, die nach dem Präfix, Objekt-Tag oder Zugriffspunkt filtert.
9. Wählen Sie Save Changes (Änderungen speichern).
10. Wählen Sie im Tab Request metrics (Anforderungsmetriken) unter Filters (Filter) den Filter aus, den Sie gerade erstellt haben.

    Nach etwa 15 Minuten CloudWatch beginnt mit der Verfolgung dieser Anforderungsmetriken. Sie können sie im Tab Request metrics (Anforderungsmetriken) sehen. Sie können Diagramme für die Metriken auf der Amazon S3- oder - CloudWatch Konsole anzeigen. Anforderungsmetriken werden zum CloudWatch Standardtarif abgerechnet. Weitere Informationen finden Sie unter Amazon- CloudWatch Preise.

Verwenden der REST-API

Mit der Amazon-S3-REST-API können Sie Metrik-Konfigurationen programmgesteuert hinzufügen. Weitere Informationen zum Hinzufügen von und Arbeiten mit Metrik-Konfigurationen finden Sie in den folgenden Themen der Amazon Simple Storage Service API Reference:

- PUT Bucket metrics
- GET Bucket-Metrikkonfiguration
- List Bucket-Metrikkonfiguration
- DELETE Bucket-Metrikkonfiguration

Verwenden der AWS CLI

1. Installieren und richten Sie die ein AWS CLI. Anleitungen finden Sie unter Installieren, Aktualisieren und Deinstallieren der AWS CLI im AWS Command Line Interface - Benutzerhandbuch.
2. Öffnen Sie ein Terminalfenster.
3. Führen Sie den folgenden Befehl aus, um eine Metrik-Konfiguration hinzuzufügen.

```
aws s3api put-bucket-metrics-configuration --endpoint https://s3.us-
west-2.amazonaws.com --bucket bucket-name --id metrics-config-id --metrics-
configuration '{"Id":"metrics-config-id"}'
```

Erstellen einer Metrik-Konfiguration, die nach dem Präfix, Objekt-Tag oder Zugriffspunkt filtert

Es gibt drei Arten von Amazon- CloudWatch Metriken für Amazon S3: Speichermetriken, Anforderungsmetriken und Replikationsmetriken. Speichermetriken werden einmal pro Tag gemeldet und allen Kunden ohne zusätzliche Kosten zur Verfügung gestellt. Die Anforderungsmetriken sind in 1-Minuten-Intervallen nach einer gewissen Latenz für die Verarbeitung verfügbar. Anforderungsmetriken werden zum CloudWatch Standardtarif abgerechnet. Sie wählen Anforderungsmetriken aus, indem Sie diese in der Konsole konfigurieren oder die Amazon-S3-API verwenden. S3-Replikationsmetriken enthalten detaillierte Metriken für die Replikationsregeln in Ihrer Replikationskonfiguration. Mit Replikationsmetriken können Sie den minute-by-minute Fortschritt überwachen, indem Sie ausstehende Bytes, ausstehende Operationen, Operationen, bei denen die Replikation fehlgeschlagen ist, und die Replikationslatenz verfolgen.

Weitere Informationen zu CloudWatch Metriken für Amazon S3 finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.

Wenn Sie CloudWatch Metriken konfigurieren, können Sie einen Filter für alle Objekte in Ihrem Bucket erstellen oder die Konfiguration in Gruppen verwandter Objekte innerhalb eines einzelnen Buckets filtern. Sie können Objekte in einem Bucket zur Aufnahme in eine Metrik-Konfiguration abhängig von einem oder mehreren der folgenden Filtertypen filtern:

- Object key name prefix (Objekt-Schlüsselnamepräfix) – Obwohl das Amazon-S3-Datenmodell eine flache Struktur aufweist, können Sie eine Hierarchie durch die Verwendung eines Präfixes inferieren. Die Amazon-S3-Konsole unterstützt diese Präfixe mit dem Ordnerkonzept. Wenn Sie nach dem Präfix filtern, werden Objekte mit demselben Präfix in die Metrik-Konfiguration aufgenommen. Weitere Informationen zu Präfixen finden Sie unter Organisieren von Objekten mit Präfixen.
- Tag – Sie können Objekten Markierungen hinzufügen, d. h. Schlüsselwert-Namenspaare. Mit Markierungen können Sie Objekte einfacher finden und organisieren. Markierungen können auch als Filter für Metrik-Konfigurationen verwendet werden. Weitere Informationen über Objekt-Markierungen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.
- Zugriffspunkt – S3-Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind und vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in S3. Wenn Sie einen Zugriffspunkt-Filter erstellen, enthält Amazon S3 Anforderungen an den Zugriffspunkt, den Sie in der Metrikkonfiguration angeben. Weitere Informationen finden Sie unter Überwachen und Protokollieren von Zugriffspunkten.

> **Note**
>
> Wenn Sie eine Metrikkonfiguration erstellen, die nach Zugriffspunkt filtert, müssen Sie den Zugriffspunkt Amazon-Ressourcenname (ARN) und nicht den Zugriffspunkt-Alias verwenden. Stellen Sie sicher, dass Sie den ARN für den Zugriffspunkt selbst verwenden, nicht den ARN für ein bestimmtes Objekt. Weitere Informationen zu Zugriffspunkt-ARNs finden Sie unter Verwenden von Zugriffspunkten.

Wenn Sie einen Filter angeben, können nur Anforderungen, die für einzelne Objekte ausgeführt werden, mit dem Filter übereinstimmen und in die gemeldeten Metriken aufgenommen werden. Anforderungen wie - DeleteObjectsund -`ListObjects`Anforderungen geben keine Metriken für Konfigurationen mit Filtern zurück.

Um ein komplexeres Filtern anzufordern, wählen Sie zwei oder mehr Elemente aus. Nur Objekte, die alle diese Elemente besitzen, werden in die Metrik-Konfiguration aufgenommen. Wenn Sie keine Filter setzen, werden alle Objekte aus dem Bucket in die Metrikenkonfiguration aufgenommen.

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets (Buckets) den Namen des Buckets mit den Objekte aus, für die Sie Anforderungsmetriken anfordern möchten.
3. Wählen Sie den Tab Metrics.
4. Wählen Sie unter Bucket metrics (Bucket-Metriken) View additional charts (Zusätzliche Diagramme anzeigen) aus.
5. Wählen Sie den Tab Request metrics (Anforderungsmetriken).
6. Wählen Sie Create Filter (Filter erstellen).
7. Geben Sie im Feld Filter name (Filtername) Ihren Filternamen ein.

   Namen dürfen nur Buchstaben, Zahlen, Punkte, Bindestriche und Unterstriche enthalten.
8. Wählen Sie unter Filterumfang Begrenzen Sie den Bereich dieses Filters mit einem Präfix, Objekt-Tags und einem S3-Zugriffspunkt oder einer Kombination aus allen drei.
9. Wählen Sie unter Filtertyp mindestens einen Filtertyp aus: Präfix,Objekt-Tags, oder Zugriffspunkt.
10. Um einen Präfixfilter zu definieren und den Bereich des Filters auf einen einzelnen Pfad zu beschränken, geben Sie im Feld Präfix ein Präfix ein.
11. Um einen Objekt-Tags-Filter zu definieren, wählen Sie unterObjekt-Tags Hinzufügen eines Tags, und geben Sie dann ein Tag Schlüssel und Wert ein.
12. Um einen Zugriffspunktsfilter zu definieren, geben Sie im Feld S3-Zugriffspunkt den Zugriffspunkt-ARN ein, oder wählen Sie S3 durchsuchen, um zum Zugriffspunkt zu navigieren.

    > **Important**
    >
    > Sie können keinen Zugriffspunkt-Alias eingeben. Sie müssen den ARN für den Zugriffspunkt selbst eingeben, nicht den ARN für ein bestimmtes Objekt.
13. Wählen Sie Änderungen speichern aus.

Amazon S3 erstellt einen Filter, der das Präfix, die Tags oder den Zugriffspunkt verwendet, den Sie angegeben haben.

14. Wählen Sie im Tab Request metrics (Anforderungsmetriken) unter Filters (Filter) den Filter aus, den Sie gerade erstellt haben.

    Sie haben nun einen Filter erstellt, der den Bereich der Anforderungsmetriken nach Präfix, Objekt-Tags oder Zugriffspunkt einschränkt. Ca. 15 Minuten, nachdem mit der Verfolgung dieser Anforderungsmetriken CloudWatch begonnen hat, können Sie Diagramme für die Metriken sowohl auf der Amazon S3- als auch auf der CloudWatch Konsole sehen. Anforderungsmetriken werden zum CloudWatch Standardtarif abgerechnet. Weitere Informationen finden Sie unter Amazon- CloudWatch Preise.

    Sie können Anforderungsmetriken auch auf Bucket-Ebene konfigurieren. Weitere Informationen finden Sie unter Erstellen einer CloudWatch Metrikkonfiguration für alle Objekte in Ihrem Bucket.

Verwenden der AWS CLI

1. Installieren und richten Sie die ein AWS CLI. Anleitungen finden Sie unter Installieren, Aktualisieren und Deinstallieren der AWS CLI im AWS Command Line Interface - Benutzerhandbuch.
2. Öffnen Sie ein Terminalfenster.
3. Führen Sie einen der folgenden Befehle aus, um eine Metrik-Konfiguration hinzuzufügen:

   Example : So filtern Sie nach Präfix

   ```
   aws s3api put-bucket-metrics-configuration --bucket DOC-EXAMPLE-BUCKET1 --
   id metrics-config-id --metrics-configuration '{"Id":"metrics-config-id", "Filter":
   {"Prefix":"prefix1"}} '
   ```

   Example : So filtern Sie nach Tags

   ```
   aws s3api put-bucket-metrics-configuration --bucket DOC-EXAMPLE-BUCKET1 --
   id metrics-config-id --metrics-configuration '{"Id":"metrics-config-id", "Filter":
   {"Tag": {"Key": "string", "Value": "string"}} '
   ```

Example : So filtern Sie nach Zugriffspunkt

```
aws s3api put-bucket-metrics-configuration --bucket DOC-EXAMPLE-BUCKET1 --
id metrics-config-id --metrics-configuration '{"Id":"metrics-config-id", "Filter":
{"AccessPointArn":"arn:aws:s3:Region:account-id:accesspoint/access-point-name"}} '
```

Example : So filtern Sie nach Präfix, Tags und Zugriffspunkt

```
aws s3api put-bucket-metrics-configuration --endpoint https://
s3.Region.amazonaws.com --bucket DOC-EXAMPLE-BUCKET1 --id metrics-config-id --
metrics-configuration '
{
    "Id": "metrics-config-id",
    "Filter": {
        "And": {
            "Prefix": "string",
            "Tags": [
                {
                    "Key": "string",
                    "Value": "string"
                }
            ],
            "AccessPointArn": "arn:aws:s3:Region:account-id:accesspoint/access-
point-name"
        }
    }
}'
```

Verwenden der REST-API

Mit der Amazon-S3-REST-API können Sie Metrik-Konfigurationen programmgesteuert hinzufügen. Weitere Informationen zum Hinzufügen von und Arbeiten mit Metrik-Konfigurationen finden Sie in den folgenden Themen der Amazon Simple Storage Service API Reference:

- PUT Bucket metrics
- GET Bucket-Metrikkonfiguration
- List Bucket-Metrikkonfiguration
- DELETE Bucket-Metrikkonfiguration

Löschen eines Metrikfilters

Sie können einen Amazon- CloudWatch Anforderungsmetrikfilter löschen, wenn Sie ihn nicht mehr benötigen. Wenn Sie einen Filter löschen, werden Ihnen keine Anforderungsmetriken mehr berechnet, die diesen spezifischen Filter verwenden. Ihnen werden jedoch weiter alle anderen vorhandenen Filterkonfigurationen berechnet.

Wenn Sie einen Filter löschen, können Sie den Filter nicht mehr für Anforderungsmetriken verwenden. Das Löschen eines Filters kann nicht rückgängig gemacht werden.

Informationen zum Erstellen eines Anforderungsmetrikfilters finden Sie in den folgenden Themen:

- Erstellen einer CloudWatch Metrikkonfiguration für alle Objekte in Ihrem Bucket
- Erstellen einer Metrik-Konfiguration, die nach dem Präfix, Objekt-Tag oder Zugriffspunkt filtert

Verwenden der S3-Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets (Buckets) den Namen Ihres Buckets.
3. Wählen Sie den Tab Metrics.
4. Wählen Sie unter Bucket metrics (Bucket-Metriken) View additional charts (Zusätzliche Diagramme anzeigen) aus.
5. Wählen Sie den Tab Request metrics (Anforderungsmetriken).
6. Wählen Sie Manage filters (Filter verwalten).
7. Wählen Sie Ihren Filter aus.

   > **Important**
   >
   > Das Löschen eines Filters kann nicht rückgängig gemacht werden.

8. Wählen Sie Delete (Löschen).

   Amazon S3 löscht Ihren Filter.

Verwenden der REST-API

Mit der Amazon-S3-REST-API können Sie Metrik-Konfigurationen programmgesteuert hinzufügen. Weitere Informationen zum Hinzufügen von und Arbeiten mit Metrik-Konfigurationen finden Sie in den folgenden Themen der Amazon Simple Storage Service API Reference:

- PUT Bucket metrics
- GET Bucket-Metrikkonfiguration
- List Bucket-Metrikkonfiguration
- DELETE Bucket-Metrikkonfiguration

Amazon-S3-Ereignis-Benachrichtigungen

Sie können die Amazon-S3-Funktion für Ereignis-Benachrichtigungen verwenden, um Benachrichtigungen zu erhalten, wenn bestimmte Ereignisse in Ihrem S3-Bucket eintreten. Um Benachrichtigungen zu aktivieren, fügen Sie eine Benachrichtigungskonfiguration hinzu, die die Ereignisse angibt, die Amazon S3 veröffentlichen soll. Stellen Sie sicher, dass es auch die Ziele angibt, an die Amazon S3 die Benachrichtigungen senden soll. Sie speichern dieses Konfiguration in der einem Bucket zugeordneten Benachrichtigung-Unterressourcen. Weitere Informationen finden Sie unter Optionen für die Bucket-Konfiguration. Amazon S3 stellt eine API für Sie bereit, mit der Sie diese Subressource verwalten können.

> **Important**
>
> Amazon-S3-Ereignis-Benachrichtigungen sind so konzipiert, dass sie mindestens einmal zugestellt werden. Ereignis-Benachrichtigungen stellen Ereignisse typischerweise in wenigen Sekunden bereit, manchmal kann dies aber auch eine Minute oder länger dauern.

Übersicht über Amazon-S3-Ereignis-Benachrichtigungen

Zurzeit kann Amazon S3 Benachrichtigungen für die folgenden Ereignisse veröffentlichen:

- Neues Objekt erstellte Ereignisse
- Ereignisse zum Entfernen von Objekten
- Wiederherstellen von Objekt-Ereignissen

- Ereignis für ein verlorenes RRS-Objekt (Reduced Redundancy Storage)
- Replikationsereignisse
- S3-Lebenszyklus-Ablaufereignisse
- S3-Lebenszyklusübergangsereignisse
- Aktivieren der automatischen S3-Intelligent-Tiering-Archivierung
- Ereignismarkierungen von Objekten
- PUT-Ereignisse der Objekt-ACL

Eine vollständige Beschreibung aller unterstützten Ereignistypen finden Sie unter Unterstützte Ereignistypen für SQS, SNS und Lambda.

Amazon S3 kann Ereignis-Benachrichtigungsmeldungen an die folgenden Ziele senden. Sie spezifizieren den Wert des Amazon-Ressourcennamens (ARN) dieser Ziele in der Benachrichtigungskonfiguration.

- Amazon Simple Notification Service (Amazon SNS)-Themen
- Amazon Simple Queue Service (Amazon SQS)-Warteschlangen
- AWS Lambda -Funktion
- Amazon EventBridge

Weitere Informationen finden Sie unter Unterstützte Ereignisziele.

> **Note**
>
> FIFO-Warteschlangen (First-In-First-Out) von Amazon Simple Queue Service werden nicht als Ziel für Amazon-S3-Ereignisbenachrichtigungen unterstützt. Um eine Benachrichtigung für ein Amazon S3-Ereignis an eine Amazon SQS-FIFO-Warteschlange zu senden, können Sie Amazon verwenden EventBridge. Weitere Informationen finden Sie unter Aktivieren von Amazon EventBridge.

> **Warning**
>
> Wenn Ihre Benachrichtigung in den selben Bucket schreibt, der die Benachrichtigung auslöst, kann dies zu einer Ausführungsschleife führen. Wenn der Bucket z. B. eine Lambda-Funktion

immer dann auslöst, wenn ein Objekt hochgeladen wird, und die Funktion ein Objekt in den Bucket hochlädt, löst sich die Funktion indirekt selbst aus. Um dies zu vermeiden, verwenden Sie zwei Buckets oder konfigurieren Sie den Auslöser so, dass er nur für einen Präfix gilt, der für eingehende Objekte verwendet wird.
Weitere Informationen und ein Beispiel für die Verwendung von Amazon S3-Benachrichtigungen mit AWS Lambda finden Sie unter Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda -Entwicklerhandbuch.

Weitere Informationen zur Anzahl der Konfigurationen für Ereignisbenachrichtigungen, die Sie pro Bucket erstellen können, finden Sie unter Amazon-S3-Service-Quotas in der Allgemeinen AWS -Referenz.

Weitere Informationen zu Ereignis-Benachrichtigungen finden Sie in den folgenden Abschnitten.

Themen

Ereignis-Benachrichtigungstypen und -ziele

Amazon S3 unterstützt verschiedene Arten von Ereignis-Benachrichtigungen und Ziele, wo die Benachrichtigungen veröffentlicht werden können. Sie können den Ereignistyp und das Ziel angeben, wenn Sie Ihre Ereignis-Benachrichtigungen konfigurieren. Für jede Ereignisbenachrichtigung kann nur ein Ziel angegeben werden. Amazon-S3-Ereignisbenachrichtigungen senden für jede Benachrichtigungsmeldung einen Ereigniseintrag.

Themen

Unterstützte Ereignisziele

Amazon S3 kann Ereignis-Benachrichtigungsmeldungen an die folgenden Ziele senden.

- Amazon Simple Notification Service (Amazon SNS)-Themen
- Amazon Simple Queue Service (Amazon SQS)-Warteschlangen
- AWS Lambda
- Amazon EventBridge

Für jede Ereignisbenachrichtigung kann jedoch nur ein Zieltyp angegeben werden.

> **Note**
>
> Sie müssen Amazon S3 die Berechtigungen erteilen, Meldungen in einem Amazon SNS-Thema oder einer Amazon SQS-Warteschlange zu veröffentlichen. Sie müssen Amazon S3 auch die Berechtigung erteilen, eine - AWS Lambda Funktion in Ihrem Namen aufzurufen. Anweisungen zum Erteilen dieser Berechtigungen finden Sie unter Erteilen von Berechtigungen zur Veröffentlichung von Ereignis-Benachrichtigungsmeldungen an einem Ziel.

Amazon SNS-Thema

Amazon SNS ist ein flexibler, vollständig verwalteter Push-Messaging-Service. Sie können diesen Dienst verwenden, um Nachrichten an mobile Geräte oder verteilte Services zu senden. Mit SNS , können Sie eine Nachricht einmal veröffentlichen und ein- oder mehrmals übermitteln. Derzeit ist nur ,Standard-SNS als S3-Ereignis-Benachrichtigungsziel zulässig, während SNS FIFO nicht zulässig ist.

Amazon SNS koordiniert und verwaltet das Senden und Zustellen von Nachrichten an abonnierte Endpunkte oder Clients. Sie können mit der Amazon SNS-Konsole ein Amazon SNS-Thema erstellen, an das Ihre Benachrichtigungen gesendet werden können.

Das Thema muss sich in derselben AWS-Region wie Ihr Amazon S3-Bucket befinden. Anweisungen zum Erstellen eines Amazon-SNS-Themas finden Sie unter Erste Schritte mit Amazon SNS im Entwicklerhandbuch zu Amazon Simple Notification Service und in den Häufig gestellten Fragen zu Amazon SNS.

Damit Sie das Amazon SNS-Thema nutzen können, das Sie als Ereignis-Benachrichtigungsziel erstellen, brauchen Sie Folgendes:

- Den Amazon-Ressourcennamen (ARN) für das Amazon SNS-Thema

- Ein gültiges Amazon-SNS-Themen-Abonnement. Damit werden Themenabonnenten benachrichtigt, wenn eine Nachricht zu Ihrem Amazon-SNS-Thema veröffentlicht wird.

Amazon SQS-Warteschlange

Amazon SQS bietet zuverlässige und skalierbare gehostete Warteschlangen zum Speichern von Nachrichten, die zwischen den Computern gesendet werden. Mit Amazon SQS können Sie beliebige Datenvolumen übertragen, ohne dass andere Services stets verfügbar sein müssen. Sie können mit der Amazon SQS-Konsole eine Amazon SQS-Warteschlange erstellen, an die Ihre Benachrichtigungen gesendet werden können.

Die Amazon SQS-Warteschlange muss sich in derselben AWS-Region wie Ihr Amazon S3Bucket befinden. Anweisungen zum Erstellen einer Amazon-SQS-Warteschlange finden Sie unter Was ist Amazon Simple Queue Service und Erste Schritte mit Amazon SQS im Entwicklerhandbuch für Amazon Simple Queue Service.

Damit Sie die Amazon SQS-Warteschlange als Ereignis-Benachrichtigungsziel nutzen können, brauchen Sie Folgendes:

- Der Amazon-Ressourcennamen (ARN) für die Amazon-SQS-Warteschlange

> **Note**
>
> FIFO-Warteschlangen (First-In-First-Out) von Amazon Simple Queue Service werden nicht als Ziel für Amazon-S3-Ereignisbenachrichtigungen unterstützt. Um eine Benachrichtigung für ein Amazon S3-Ereignis an eine Amazon SQS-FIFO-Warteschlange zu senden, können Sie Amazon verwenden EventBridge. Weitere Informationen finden Sie unter Aktivieren von Amazon EventBridge.

Lambda-Funktion

Sie können verwenden, AWS Lambda um andere - AWS Services mit benutzerdefinierter Logik zu erweitern, oder Ihr eigenes Backend erstellen, das in AWS großem Umfang, Leistung und Sicherheit arbeitet. Mit Lambda können Sie diskrete ereignisabhängige Anwendungen erstellen, die nur ausgeführt werden, wenn sie benötigt werden. Sie können damit auch diese Anwendungen automatisch von ein paar Anfragen pro Tag auf Tausende pro Sekunde skalieren.

Lambda kann benutzerdefinierten Code in Reaktion auf Amazon-S3-Bucket-Ereignisse ausführen. Sie laden Ihren benutzerdefinierten Code auf Lambda hoch und erstellen eine sogenannte Lambda-Funktion. Wenn Amazon S3 ein Ereignis eines bestimmten Typs erkennt, kann es das Ereignis in veröffentlichen AWS Lambda und Ihre Funktion in Lambda aufrufen. Lambda führt als Antwort Ihre Funktion aus. Ein Ereignistyp, den er möglicherweise erkennt, ist beispielsweise ein objekterstelltes Ereignis.

Sie können die AWS Lambda Konsole verwenden, um eine Lambda-Funktion zu erstellen, die die AWS Infrastruktur verwendet, um den Code in Ihrem Namen auszuführen. Die Lambda-Funktion muss sich in derselben Region wie Ihr S3-Bucket befinden. Sie müssen auch den Namen oder den ARN einer Lambda-Funktion haben, um die Lambda-Funktion als Ziel für Ereignis-Benachrichtigungen einzurichten.

> **Warning**
>
> Wenn Ihre Benachrichtigung in den selben Bucket schreibt, der die Benachrichtigung auslöst, kann dies zu einer Ausführungsschleife führen. Wenn der Bucket z. B. eine Lambda-Funktion immer dann auslöst, wenn ein Objekt hochgeladen wird, und die Funktion ein Objekt in den Bucket hochlädt, löst sich die Funktion indirekt selbst aus. Um dies zu vermeiden, verwenden Sie zwei Buckets oder konfigurieren Sie den Auslöser so, dass er nur für einen Präfix gilt, der für eingehende Objekte verwendet wird.
> Weitere Informationen und ein Beispiel für die Verwendung von Amazon S3-Benachrichtigungen mit AWS Lambda finden Sie unter Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda -Entwicklerhandbuch.

Amazon EventBridge

Amazon EventBridge ist ein Serverless Event Bus, der Ereignisse von - AWS Services empfängt. Sie können Regeln einrichten, um Ereignisse abzugleichen und sie an Ziele wie einen AWS -Service oder einen HTTP-Endpunkt zu übermitteln. Weitere Informationen finden Sie unter Was ist EventBridge im Amazon- EventBridge Benutzerhandbuch.

Im Gegensatz zu anderen Zielen können Sie Ereignisse aktivieren oder deaktivieren, die EventBridge für einen Bucket an übermittelt werden sollen. Wenn Sie die Zustellung aktivieren, werden alle Ereignisse an gesendet EventBridge. Darüber hinaus können Sie EventBridge Regeln verwenden, um Ereignisse an zusätzliche Ziele weiterzuleiten.

Unterstützte Ereignistypen für SQS, SNS und Lambda

Amazon S3 kann Ereignisse der folgenden Typen veröffentlichen. Sie spezifizieren diese Ereignistypen in der Benachrichtigungskonfiguration.

Ereignistypen	Beschreibung
s3:TestEvent	Wenn eine Benachrichtigung aktiviert ist, veröffentlicht Amazon S3 eine Testbenachrichtigung. Damit soll sichergestellt werden, dass das Thema vorhanden ist und dass der Bucket-Eigentümer über die Berechtigung verfügt, das angegebene Thema zu veröffentlichen. Wenn das Aktivieren der Benachrichtigung fehlschlägt, erhalten Sie keine Testbenachrichtigung.
s3:ObjectCreated:* s3:ObjectCreated:Put s3:ObjectCreated:Post s3:ObjectCreated:Copy s3:ObjectCreated:CompleteMultipartUpload	Amazon-S3-API-Operationen wie PUT, POST und COPY können ein Objekt erstellen. Mit diesen Ereignistypen können Sie Benachrichtigungen aktivieren, wenn ein Objekt mit einer bestimmten API-Operation erstellt wird. Sie können auch den `s3:ObjectCreated:*` Ereignistyp verwenden, um eine Benachrichtigung unabhängig von der API anfordern, die zum Erstellen eines Objekts verwendet wurde. `s3:ObjectCreated:CompleteMultipartUpload` enthält Objekte, die mit UploadPartCopy für Kopiervorgänge erstellt werden.
s3:ObjectRemoved:* s3:ObjectRemoved:Delete s3:ObjectRemoved:DeleteMarkerCreated	Mithilfe der ObjectRemoved Ereignistypen können Sie Benachrichtigungen aktivieren, wenn ein Objekt oder ein Objektstapel aus einem Bucket entfernt wird. Sie können eine Benachrichtigung anfordern, wenn ein Objekt gelöscht wird, oder wenn ein versionsfähiges Objekt permanent gelöscht wird. Dazu wird der Ereignistyp `s3:ObjectRemoved:Delete` verwendet. Alternativ können Sie eine Benachrichtigung anfordern, wenn eine Löschmarkierung für ein versioniertes Objekt mit

Ereignistypen	Beschreibung
	`s3:ObjectRemoved:DeleteMarkerCreated` erstellt wird. Eine entsprechende Anleitung zum Löschen von versionierten Objekten finden Sie unter Löschen von Objekten aus einem versioning-fähigen Bucket. Sie können auch eine Wildcard `s3:ObjectRemoved:*` verwenden , um bei jedem Löschen eines Objekts eine Benachric htigung anzufordern. Diese Ereignisbenachrichtigungen warnen Sie nicht vor automatischen Löschungen aufgrund von Lebenszyk luskonfigurationen oder fehlgeschlagenen Operationen.
s3:ObjectRestore:* s3:ObjectRestore:Post s3:ObjectRestore:Abgeschlossen s3:ObjectRestore:Delete	Durch die Verwendung der ObjectRestore Ereignistypen können Sie beim Wiederherstellen von Objekten aus der Speicherklasse S3 Glacier Flexible Retrieval, der Speicherk lasse S3 Glacier Deep Archive, der Stufe S3 Intelligent-Tiering Archive Access und der Stufe S3 Intelligent-Tierin g Deep Archive Access Benachrichtigungen zur Ereignisa uslösung und zum Abschluss erhalten. Sie können auch Benachrichtigungen erhalten, wann die wiederhergestellte Kopie eines Objekts abläuft. Der `s3:ObjectRestore:Post` -Ereignistyp informiert Sie über die Initiierung von Objektwiederherstellungen. Der `s3:ObjectRestore:Completed` -Ereignistyp informier t Sie über den Abschluss von Wiederherstellungen. Der `s3:ObjectRestore:Delete` -Ereignistyp benachric htigt Sie, wenn die temporäre Kopie eines wiederher gestellten Objekts abläuft.
s3:ReducedRedundancyLostObject	Sie erhalten dieses Benachrichtigungsereignis, wenn Amazon S3 erkennt, dass ein Objekt der RRS-Speic herklasse verloren geht.

Ereignistypen	Beschreibung
s3:Replikation:* s3:Replikation:OperationFailedReplication s3:Replikation:OperationMissedThreshold s3:Replikation:OperationReplicatedAfterThreshold s3:Replikation:OperationNotTracked	Mithilfe der Replikationsereignistypen können Sie Benachrichtigungen für Replikationskonfigurationen erhalten, bei denen S3-Replikationsmetriken oder S3-Replikationszeitsteuerung (S3 RTC) aktiviert sind. Sie können den minute-by-minute Fortschritt von Replikationsereignissen überwachen, indem Sie ausstehende Bytes, ausstehende Operationen und Replikationslatenz verfolgen. Informationen zu Replikationsmetriken finden Sie unter Überwachen des Fortschritts mit Replikationsmetriken und S3-Ereignisbenachrichtigungen. Der `s3:Replication:OperationFailedReplication` -Ereignistyp benachrichtigt Sie, wenn ein Objekt, das für die Replikation berechtigt war, nicht repliziert werden konnte. Der `s3:Replication:OperationMissedThreshold` -Ereignistyp benachrichtigt Sie, wenn ein Objekt, das für die Replikation berechtigt war, den 15-minütigen Schwellenwert für die Replikation überschreitet. Der `s3:Replication:OperationReplicatedAfterThreshold` -Ereignistyp benachrichtigt Sie, wenn ein Objekt, bei dem eine Replikation mithilfe von S3 Replication Time Control zulässig war, und das nach dem 15-minütigen Schwellenwert repliziert wird. Der `s3:Replication:OperationNotTracked` -Ereignistyp benachrichtigt Sie, wenn ein Objekt, das für die Replikation geeignet war und die S3-Replikationszeitsteuerung verwendet, aber nicht mehr von Replikationsmetriken verfolgt wird.

Ereignistypen	Beschreibung
s3:LifecycleExpiration:* s3:LifecycleExpiration:Delete s3:LifecycleExpiration:DeleteMarkerCreated	Durch die Verwendung der LifecycleExpiration Ereignistypen können Sie eine Benachrichtigung erhalten, wenn Amazon S3 ein Objekt basierend auf Ihrer S3-Lebenszykluskonfiguration löscht. Der Ereignistyp `s3:LifecycleExpiration:Delete` benachrichtigt Sie, wenn ein Objekt in einem unversionierten Bucket gelöscht wird. Es benachrichtigt Sie auch, wenn eine Objektversion durch eine S3-Lebenszyklus-Konfiguration dauerhaft gelöscht wird. Der Ereignistyp `s3:LifecycleExpiration:DeleteMarkerCreated` benachrichtigt Sie, wenn S3 Lebenszyklus eine Löschmarke erstellt, wenn eine aktuelle Version eines Objekts im versionierten Bucket gelöscht wird.
s3:LifecycleTransition	Sie erhalten dieses Benachrichtigungsereignis, wenn ein Objekt von einer S3-Lebenszykluskonfiguration in eine andere Amazon-S3-Speicherklasse überführt wird.
s3:IntelligentTiering	Sie erhalten dieses Benachrichtigungsereignis, wenn ein Objekt innerhalb der Speicherklasse S3 Intelligent-Tiering in die Stufe Archive Access oder Deep Archive Access verschoben wird.
s3:ObjectTagging:* s3:ObjectTagging:Put s3:ObjectTagging:Delete	Mithilfe der ObjectTagging Ereignistypen können Sie Benachrichtigungen aktivieren, wenn ein Objekt-Tag zu einem Objekt hinzugefügt oder daraus gelöscht wird. Der `s3:ObjectTagging:Put` -Ereignistyp benachrichtigt Sie, wenn ein Tag für ein Objekt PUT ist oder ein vorhandenes Tag aktualisiert wird. Der `s3:ObjectTagging:Delete` -Ereignistyp benachrichtigt Sie, wenn ein Tag aus einem Objekt entfernt wird.

Ereignistypen	Beschreibung
s3:ObjectAcl:Put	Sie erhalten dieses Benachrichtigungsereignis, wenn eine ACL für ein Objekt PUT ist oder wenn eine vorhanden e ACL geändert wird. Ein Ereignis wird nicht generiert, wenn eine Anforderung keine Änderung an der ACL eines Objekts zur Folge hat.

Unterstützte Ereignistypen für Amazon EventBridge

Eine Liste der Ereignistypen, die Amazon S3 an Amazon sendet EventBridge, finden Sie unter Verwenden von EventBridge.

Verwenden von Amazon SQS, Amazon SNS und Lambda

Die Aktivierung von Benachrichtigungen erfolgt auf Bucket-Ebene. Sie speichern Informationen zur Benachrichtigungskonfiguration in der Benachrichtigungen-Unterressourcen die einem Bucket zugeordnet sind. Nachdem Sie die Bucket-Benachrichtigungskonfiguration erstellt oder geändert haben, dauert es normalerweise etwa fünf Minuten, bis die Änderungen wirksam werden. Ein `s3:TestEvent` tritt auf, wenn die Benachrichtigung zum ersten Mal aktiviert wird. Sie können jede der folgenden Methoden verwenden, um die Benachrichtigungskonfiguration zu verwalten:

- Verwenden der Amazon-S3-Konsole – Die Benutzeroberfläche der Konsole ermöglicht Ihnen, eine Benachrichtigungskonfiguration für einen Bucket einzurichten, ohne Code schreiben zu müssen. Weitere Informationen finden Sie unter Aktivieren und Konfigurieren von Ereignis-Benachrichtigungen mit der Amazon-S3-Konsole.
- Programmgesteuerte Verwendung der AWS SDKs – Intern rufen sowohl die Konsole als auch die SDKs die Amazon S3-REST-API auf, um die dem Bucket zugeordneten Benachrichtigungsunterressourcen zu verwalten. Beispiele für Benachrichtigungskonfigurationen, die AWS SDK verwenden, finden Sie unter Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema oder SQS-Warteschlange).

> **Note**
>
> Sie können auch die REST-API-Aufrufe in Amazon S3 direkt von Ihrem Code aus durchführen. Dies kann jedoch umständlich sein, da Sie dazu Code schreiben müssen, um Ihre Anforderungen zu authentifizieren.

Unabhängig von der verwendeten Methode speichert Amazon S3 die Benachrichtigungskonfiguration als XML in der mit dem jeweiligen Bucket verknüpften Unterressourcen Benachrichtigungen. Weitere Informationen zu Bucket-Subressourcen finden Sie unter Optionen für die Bucket-Konfiguration.

Themen

- Erteilen von Berechtigungen zur Veröffentlichung von Ereignis-Benachrichtigungsmeldungen an einem Ziel
- Aktivieren und Konfigurieren von Ereignis-Benachrichtigungen mit der Amazon-S3-Konsole
- Programmgesteuerte Konfiguration von Ereignis-Benachrichtigungen
- Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema oder SQS-Warteschlange)
- Konfigurieren von Ereignis-Benachrichtigungen mithilfe der Namensfilterung
- Struktur von Ereignismeldungen

Erteilen von Berechtigungen zur Veröffentlichung von Ereignis-Benachrichtigungsmeldungen an einem Ziel

Sie müssen dem Amazon-S3-Prinzipal die erforderlichen Berechtigungen erteilen, um die relevante API aufzurufen, um Nachrichten in einem SNS-Thema, einer SQS-Warteschlange oder einer Lambda-Funktion zu veröffentlichen. So kann Amazon S3 Ereignisbenachrichtigungsmeldungen an einem Ziel veröffentlichen.

Informationen zur Fehlerbehebung beim Veröffentlichen von Ereignisbenachrichtigungen in einem Ziel finden Sie unter Fehlerbehebung bei der Veröffentlichung von Amazon-S3-Ereignisbenachrichtigungen in einem Thema von Amazon Simple Notification Service.

Themen

- Erteilen von Berechtigungen zum Aufrufen einer - AWS Lambda Funktion

- Erteilen von Berechtigungen, Meldungen in einem SNS-Thema oder einer SQS-Warteschlange zu veröffentlichen

Erteilen von Berechtigungen zum Aufrufen einer - AWS Lambda Funktion

Amazon S3 veröffentlicht Ereignismeldungen in , AWS Lambda indem es eine Lambda-Funktion aufruft und die Ereignismeldung als Argument bereitstellt.

Wenn Sie die Amazon-S3-Konsole verwenden, um Ereignisbenachrichtigungen in einem Amazon-S3-Bucket für eine Lambda-Funktion zu konfigurieren, richtet die Konsole die erforderlichen Berechtigungen für die Lambda-Funktion ein. Dies ist so, dass Amazon S3 über Berechtigungen verfügt, die Funktion aus dem Bucket aufzurufen. Weitere Informationen finden Sie unter Aktivieren und Konfigurieren von Ereignis-Benachrichtigungen mit der Amazon-S3-Konsole.

Sie können Amazon S3 auch Berechtigungen von erteilen AWS Lambda , um Ihre Lambda-Funktion aufzurufen. Weitere Informationen finden Sie unter Tutorial: Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda -Entwicklerhandbuch.

Erteilen von Berechtigungen, Meldungen in einem SNS-Thema oder einer SQS-Warteschlange zu veröffentlichen

Um Amazon S3 Berechtigungen zum Veröffentlichen von Nachrichten im SNS-Thema oder in der SQS-Warteschlange zu erteilen, fügen Sie dem Ziel-SNS-Thema oder der SQS-Warteschlange eine AWS Identity and Access Management (IAM)-Richtlinie hinzu.

Ein Beispiel dafür, wie Sie einem SNS-Thema oder einer SQS-Warteschlange eine Richtlinie anfügen, finden Sie unter Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema oder SQS-Warteschlange). Weitere Informationen über Berechtigungen finden Sie in den folgenden Themen:

- Beispielfälle für die Amazon SNS-Zugriffssteuerung im Amazon Simple Notification Service-Entwicklerhandbuch
- Identity and Access Management in Amazon SQS im Amazon Simple Queue Service-Entwicklerhandbuch

IAM-Richtlinie für ein SNS-Zielthema

Im Folgenden finden Sie ein Beispiel für eine AWS Identity and Access Management (IAM)-Richtlinie, die Sie dem SNS-Zielthema anfügen. Anweisungen zur Verwendung dieser Richtlinie zum Einrichten

eines Amazon-SNS-Zielthemas für Ereignisbenachrichtigungen finden Sie unter Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema oder SQS-Warteschlange).

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "Example SNS topic policy",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SNS:Publish"
            ],
            "Resource": "SNS-topic-ARN",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:bucket-name"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
        }
    ]
}
```

IAM-Richtlinie für eine SQS-Zielwarteschlange

Im Folgenden finden Sie ein Beispiel für eine IAM-Richtlinie, die Sie an die SQS-Zielwarteschlange anfügen. Anweisungen zur Verwendung dieser Richtlinie zum Einrichten einer Amazon-SQS-Zielwarteschlange für Ereignisbenachrichtigungen finden Sie unter Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema oder SQS-Warteschlange).

Um diese Richtlinie verwenden zu können, müssen Sie den ARN der Amazon SQS-Warteschlange, den Bucket-Namen und die AWS-Konto ID des Bucket-Eigentümers aktualisieren.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
```

```
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SQS:SendMessage"
            ],
            "Resource": "arn:aws:sqs:Region:account-id:queue-name",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:awsexamplebucket1"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
        }
    ]
}
```

Für die IAM-Richtlinien für Amazon SNS und Amazon SQS kann die `StringLike`-Bedingung statt der `ArnLike`-Bedingung in der Richtlinie angegeben werden.

Wenn `ArnLike` verwendet wird, müssen die Teile partition, service, account-id, resource-type und teilweise resource-id-Teile des ARN genau mit dem ARN im Anforderungskontext übereinstimmen. Nur die Region und der Ressourcenpfad lassen einen teilweisen Abgleich zu.

Wenn `StringLike` anstelle von `ArnLike` verwendet wird, ignoriert der Abgleich die ARN-Struktur und ermöglicht einen teilweisen Abgleich, unabhängig davon, welcher Teil mit einem Platzhalter versehen wurde. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

```
"Condition": {
  "StringLike": { "aws:SourceArn": "arn:aws:s3:*:*:bucket-name" }
  }
```

AWS KMS Schlüsselrichtlinie

Wenn die SQS-Warteschlange oder SNS-Themen mit einem AWS Key Management Service (AWS KMS) vom Kunden verwalteten Schlüssel verschlüsselt sind, müssen Sie dem Amazon S3-Serviceprinzipal die Berechtigung erteilen, mit den verschlüsselten Themen oder der Warteschlange zu arbeiten. Um dem Amazon-S3-Service-Prinzipal die Berechtigung zu erteilen, fügen Sie der Schlüsselrichtlinie für den vom Kunden verwalteten Schlüssel die folgende Anweisung hinzu.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

Weitere Informationen zu AWS KMS Schlüsselrichtlinien finden Sie unter Verwenden von Schlüsselrichtlinien in AWS KMS im AWS Key Management Service -Entwicklerhandbuch.

Weitere Informationen zur Verwendung der serverseitigen Verschlüsselung mit AWS KMS für Amazon SQS und Amazon SNS finden Sie im Folgenden:

- Schlüsselverwaltung im Amazon Simple Notification Service-Entwicklerhandbuch.
- Schlüsselverwaltung im Amazon Simple Queue Service-Entwicklerhandbuch.
- Encrypting messages published to Amazon SNS with AWS KMS im AWS -Compute-Blog.

Aktivieren und Konfigurieren von Ereignis-Benachrichtigungen mit der Amazon-S3-Konsole

Sie können bestimmte Amazon-S3-Bucket-Ereignisse aktivieren, um eine Benachrichtigungsmeldung an ein Ziel zu senden, wenn das Ereignis auftritt. In diesem Abschnitt erfahren Sie, wie Sie mit der Amazon-S3-Konsole Ereignis-Benachrichtigungen aktivieren können. Informationen zur Verwendung von Ereignisbenachrichtigungen mit den - AWS SDKs und den Amazon S3-REST-APIs finden Sie unter Programmgesteuerte Konfiguration von Ereignis-Benachrichtigungen.

Voraussetzungen: Bevor Sie Ereignis-Benachrichtigungen für Ihren Bucket aktivieren können, müssen Sie einen der Zieltypen einrichten und dann die Berechtigungen konfigurieren. Weitere Informationen finden Sie unter Unterstützte Ereignisziele und Erteilen von Berechtigungen zur Veröffentlichung von Ereignis-Benachrichtigungsmeldungen an einem Ziel.

> **Note**
>
> FIFO-Warteschlangen (First-In-First-Out) von Amazon Simple Queue Service werden nicht als Ziel für Amazon-S3-Ereignisbenachrichtigungen unterstützt. Um eine Benachrichtigung für ein Amazon S3-Ereignis an eine Amazon SQS-FIFO-Warteschlange zu senden, können Sie Amazon verwenden EventBridge. Weitere Informationen finden Sie unter Aktivieren von Amazon EventBridge.

Themen

Aktivieren von Amazon-SNS-, Amazon-SQS- oder Lambda-Benachrichtigungen über die Amazon-S3-Konsole

Ereignis-Benachrichtigungen für einen S3-Bucket aktivieren und konfigurieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie Ereignisse aktivieren möchten.
3. Wählen Sie Properties (Eigenschaften).

4. Navigieren Sie zum Abschnitt Event Notifications (Ereignis-Benachrichtigungen) und wählen Sie Create event notification (Ereignis-Benachrichtigung erstellen)
5. Geben Sie im Abschnitt General configuration (Allgemeine Konfiguration) den beschreibenden Ereignisnamen für Ihre Ereignis-Benachrichtigung an. Optional können Sie auch ein Präfix und ein Suffix angeben, um die Benachrichtigungen auf Objekte mit Schlüsseln zu beschränken, die mit den angegebenen Zeichen enden.

   a. Geben Sie eine Beschreibung für den Namen der Veranstaltung ein.

      Wenn Sie keinen Namen eingeben, wird ein eindeutiger Bezeichner (Globally Unique Identifier, GUID) generiert und für den Namen verwendet.

   b. (Optional) Um Ereignisbenachrichtigungen nach Präfix zu filtern, geben Sie ein Präfix ein.

      Sie können beispielsweise einen Präfix-Filter einrichten, damit Sie nur Benachrichtigungen erhalten, wenn Dateien zu einem bestimmten Ordner (z. B.) hinzugefügt wer, `images/`).

   c. (Optional) Um Ereignisbenachrichtigungen nach Suffix zu filtern, geben Sie ein Suffix ein.

      Weitere Informationen finden Sie unter Konfigurieren von Ereignis-Benachrichtigungen mithilfe der Namensfilterung.

6. Wählen Sie im Abschnitt Event types (Ereignistypen) einen oder mehrere Ereignistypen aus, für die Sie Benachrichtigungen erhalten möchten.

   Eine Liste der verschiedenen Ereignistypen finden Sie unter Unterstützte Ereignistypen für SQS, SNS und Lambda.

7. Wählen Sie im Abschnitt Destination (Ziel) das Ziel für Ereignis-Benachrichtigungen aus.

   > **Note**
   >
   > Bevor Sie Ereignisbenachrichtigungen veröffentlichen können, müssen Sie dem Amazon-S3-Prinzipal die erforderlichen Berechtigungen zum Aufrufen der entsprechenden API erteilen. Dies ist so, dass Benachrichtigungen für eine Lambda-Funktion, ein SNS-Thema oder eine SQS-Warteschlange veröffentlicht werden können.

   a. Wählen Sie den Zieltyp aus: Lambda Function (Lambda-Funktion), SNS Topic (SNS-Thema), oder SQS Queue (SQS-Warteschlange).

b. Nachdem Sie Ihren Zieltyp ausgewählt haben, wählen Sie eine Funktion, ein Thema oder eine Warteschlange aus der Liste aus.

c. Wenn Sie lieber einen Amazon-Ressourcennamen (ARN) angeben möchten, wählen Sie die Option Enter ARN (ARN eingeben) aus und geben Sie den ARN ein.

Weitere Informationen finden Sie unter Unterstützte Ereignisziele.

8. Wählen Sie Save changes (Änderungen speichern) und Amazon S3 sendet eine Testnachricht an das Ziel für Ereignis-Benachrichtigungen.

Programmgesteuerte Konfiguration von Ereignis-Benachrichtigungen

Standardmäßig sind für keinen Ereignistyp Benachrichtigungen aktiviert. Aus diesem Grund speichert die Benachrichtigung-Unterressourcen anfänglich eine leere Konfiguration.

```
<NotificationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</NotificationConfiguration>
```

Um die Benachrichtigungen für bestimmte Ereignistypen zu aktivieren, ersetzen Sie das XML durch die entsprechende Konfiguration, die die Ereignistypen identifiziert, die Amazon S3 veröffentlichen soll, sowie das Ziel, in dem die Ereignisse veröffentlicht werden sollen. Sie müssen für jedes Ziel eine entsprechende XML-Konfiguration hinzufügen.

So werden Ereignismeldungen in einer SQS-Warteschlange veröffentlicht

Um eine SQS-Warteschlange als Benachrichtigungsziel für einen oder mehrere Ereignistypen festzulegen, fügen Sie `QueueConfiguration` hinzu.

```
<NotificationConfiguration>
  <QueueConfiguration>
    <Id>optional-id-string</Id>
    <Queue>sqs-queue-arn</Queue>
    <Event>event-type</Event>
    <Event>event-type</Event>
     ...
  </QueueConfiguration>
   ...
</NotificationConfiguration>
```

So werden Ereignismeldungen in einem SNS-Thema veröffentlicht

Um ein SNS-Thema als Benachrichtigungsziel für bestimmte Ereignistypen festzulegen, fügen Sie `TopicConfiguration` hinzu.

```
<NotificationConfiguration>
  <TopicConfiguration>
     <Id>optional-id-string</Id>
     <Topic>sns-topic-arn</Topic>
     <Event>event-type</Event>
     <Event>event-type</Event>
      ...
  </TopicConfiguration>
   ...
</NotificationConfiguration>
```

So rufen Sie die AWS Lambda Funktion auf und geben eine Ereignisnachricht als Argument an

Um eine Lambda-Funktion als Benachrichtigungsziel für bestimmte Ereignistypen festzulegen, fügen Sie `CloudFunctionConfiguration` hinzu.

```
<NotificationConfiguration>
  <CloudFunctionConfiguration>
     <Id>optional-id-string</Id>
     <CloudFunction>cloud-function-arn</CloudFunction>
     <Event>event-type</Event>
     <Event>event-type</Event>
      ...
  </CloudFunctionConfiguration>
   ...
</NotificationConfiguration>
```

So entfernen Sie alle in einem Bucket konfigurierten Benachrichtigungen

Um alle für einen Bucket konfigurierten Benachrichtigungen zu entfernen, speichern Sie ein leeres `<NotificationConfiguration/>`-Element in der notification-Subressource.

Wenn Amazon S3 ein Ereignis des spezifischen Typs erkennt, veröffentlicht es eine Meldung mit der Ereignisinformation. Weitere Informationen finden Sie unter Struktur von Ereignismeldungen.

Weitere Informationen zur Konfiguration von Ereignis-Benachrichtigungen finden Sie in den folgenden Themen:

- Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema oder SQS-Warteschlange).
- Konfigurieren von Ereignis-Benachrichtigungen mithilfe der Namensfilterung

Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema oder SQS-Warteschlange)

Sie können Amazon-S3-Benachrichtigungen über den Amazon Simple Notification Service (Amazon SNS) oder den Amazon Simple Queue Service (Amazon SQS) erhalten. In diesem Walkthrough fügen Sie Ihrem Bucket eine Benachrichtigungskonfiguration mit einem Amazon SNS-Thema und einer Amazon SQS-Warteschlange hinzu.

> **Note**
>
> FIFO-Warteschlangen (First-In-First-Out) von Amazon Simple Queue Service werden nicht als Ziel für Amazon-S3-Ereignisbenachrichtigungen unterstützt. Um eine Benachrichtigung für ein Amazon S3-Ereignis an eine Amazon SQS-FIFO-Warteschlange zu senden, können Sie Amazon verwenden EventBridge. Weitere Informationen finden Sie unter Aktivieren von Amazon EventBridge.

Themen

- Walkthrough-Übersicht
- Schritt 1: Erstellen einer Amazon SQS-Warteschlange
- Schritt 2: Erstellen eines Amazon SNS-Themas
- Schritt 3: Hinzufügen einer Benachrichtigungskonfiguration zu Ihrem Bucket
- Schritt 4: Testen der Einrichtung

Walkthrough-Übersicht

Dieser Walkhrough hilft Ihnen bei folgenden Aufgaben:

- Veröffentlichung von Ereignissen des Typs `s3:ObjectCreated:*` in einer Amazon SQS-Warteschlange.
- Veröffentlichung von Ereignissen des Typs `s3:ReducedRedundancyLostObject` in einem Amazon SNS-Thema.

Weitere Informationen zur Benachrichtigungskonfiguration finden Sie unter Verwenden von Amazon SQS, Amazon SNS und Lambda

Alle diese Schritte können Sie auf der Konsole erledigen, ohne Code schreiben zu müssen. Darüber hinaus werden Codebeispiele bereitgestellt, die AWS SDKs für Java und .NET verwenden, um Ihnen zu helfen, Benachrichtigungskonfigurationen programmgesteuert hinzuzufügen.

In dieser Vorgehensweise werden die folgenden Schritte beschrieben:

1. Erstellen einer Amazon SQS-Warteschlange

   Mit der Amazon-SQS-Konsole erstellen Sie eine SQS-Warteschlange. Sie können auf alle Meldungen zugreifen, die Amazon S3 programmgesteuert an die Warteschlange sendet. Für diese schrittweise Anleitung überprüfen Sie die Benachrichtigungsmitteilungen in der Konsole.

   Sie ordnen der Warteschlange eine Zugriffsrichtlinie zu, um Amazon S3 die Berechtigung zu erteilen, Meldungen zu veröffentlichen.
2. Erstellen Sie ein Amazon SNS-Thema.

   Erstellen Sie mit der Amazon-SNS-Konsole ein SNS-Thema und abonnieren Sie das Thema. Auf diese Weise werden alle darin veröffentlichten Ereignisse an Sie weitergegeben. Sie geben als Kommunikationsprotokoll E-Mail an. Nachdem Sie ein Thema erstellt haben, sendet Amazon SNS eine E-Mail. Sie verwenden den Link in der E-Mail, um das Abonnement des Themas zu bestätigen.

   Sie ordnen dem Thema eine Zugriffsrichtlinie zu, um Amazon S3 die Berechtigung zu erteilen, Meldungen zu veröffentlichen.
3. Fügen Sie einem Bucket eine Benachrichtigungskonfiguration hinzu.

Schritt 1: Erstellen einer Amazon SQS-Warteschlange

Führen Sie die Schritte zum Erstellen und Abonnieren einer Amazon Simple Queue Service (Amazon SQS)-Warteschlange (Amazon SQS) aus.

1. Erstellen Sie mit der Amazon SQS-Konsole eine Warteschlange. Anweisungen finden Sie unter Erste Schritte mit Amazon SQS im Amazon Simple Queue Service-Entwicklerhandbuch.
2. Ersetzen Sie die der Warteschlange zugeordnete Zugriffsrichtlinie durch die folgende Richtlinie.

a. Wählen Sie in der Amazon-SQS-Konsole in der Liste Warteschlangen den Warteschlangennamen aus.

b. Wählen Sie im Tab Zugriffsrichtlinie die Option Bearbeiten aus.

c. Ersetzen Sie die Zugriffsrichtlinie, die der Warteschlange angefügt ist. Geben Sie darin Ihren Amazon-SQS-ARN, den Quell-Bucket-Namen und die Bucket-Eigentümer-Konto-ID an.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SQS:SendMessage"
            ],
            "Resource": "SQS-queue-ARN",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:awsexamplebucket1"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
        }
    ]
}
```

d. Wählen Sie Speichern.

3. (Optional) Wenn es sich bei der Amazon SQS-Warteschlange oder dem Amazon SNS-Thema um eine serverseitige Verschlüsselung handelt, die mit AWS Key Management Service (AWS KMS) aktiviert ist, fügen Sie die folgende Richtlinie zum zugehörigen vom Kunden verwalteten Schlüssel für symmetrische Verschlüsselung hinzu.

   Sie müssen die Richtlinie einem von Kunden verwalteten Schlüssel hinzufügen, da Sie den AWS -verwalteten Schlüssel für Amazon SQS oder Amazon SNS nicht ändern können.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

Weitere Informationen zur Verwendung von SSE für Amazon SQS und Amazon SNS mit AWS KMS finden Sie hier:

- Schlüsselverwaltung im Amazon Simple Notification Service-Entwicklerhandbuch.
- Schlüsselverwaltung im Amazon Simple Queue Service-Entwicklerhandbuch.

4. Notieren Sie den ARN der Warteschlange.

   Die SQS-Warteschlange, die Sie erstellt haben, ist eine weitere Ressource in Ihrem AWS-Konto. Es hat einen eindeutigen Amazon-Ressourcennamen (ARN). Sie benötigen diesen ARN im nächsten Schritt. Der ARN muss das folgende Format aufweisen:

   ```
   arn:aws:sqs:aws-region:account-id:queue-name
   ```

Schritt 2: Erstellen eines Amazon SNS-Themas

Gehen Sie wie folgt vor, um ein Amazon SNS-Thema zu erstellen und zu abonnieren.

1. Erstellen Sie mit der Amazon SNS-Konsole ein Thema. Eine Anleitung finden Sie unter Amazon SNS-Thema anlegen im Amazon Simple Notification Service-Entwicklerhandbuch.

2. Abonnieren Sie das Thema. Für diese Übung geben Sie email als Kommunikationsprotokoll an. Eine Anleitung finden Sie unter Amazon SNS-Thema abonnieren im Amazon Simple Notification Service-Entwicklerhandbuch.

   Sie erhalten eine E-Mail, in der Sie aufgefordert werden, das Abonnement des Themas zu bestätigen. Bestätigen Sie das Abonnement.

3. Ersetzen Sie die dem Thema zugeordnete Zugriffsrichtlinie durch die folgende Richtlinie. Geben Sie darin Ihren SNS-Themen-ARN, den Bucket-Namen und die Konto-ID des Bucket-Eigentümers an.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "Example SNS topic policy",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SNS:Publish"
            ],
            "Resource": "SNS-topic-ARN",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:bucket-name"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
        }
    ]
}
```

4. Notieren Sie den ARN des Themas.

   Das von Ihnen erstellte SNS-Thema ist eine weitere Ressource in Ihrem und hat einen AWS-Konto eindeutigen ARN. Sie benötigen diesen ARN im nächsten Schritt. Der ARN hat das folgende Format:

```
arn:aws:sns:aws-region:account-id:topic-name
```

Schritt 3: Hinzufügen einer Benachrichtigungskonfiguration zu Ihrem Bucket

Sie können Bucket-Benachrichtigungen entweder über die Amazon S3-Konsole oder programmgesteuert über AWS SDKs aktivieren. Wählen Sie eine der Optionen für die Konfiguration von Benachrichtigungen über Ihren Bucket. Dieser Abschnitt zeigt Beispiele für die Verwendung des AWS -SDKs for Java und .NET.

Option A: Aktivieren von Benachrichtigungen über einen Bucket unter Verwendung der Konsole

Fügen Sie mithilfe der Amazon-S3-Konsole eine Benachrichtigungskonfiguration hinzu, die Amazon S3 zu Folgendem auffordert:

- Veröffentlichen von Ereignissen des Typs All object create events (Alle Objekterstellungsereignisse) in Ihrer Amazon SQS-Warteschlange.
- Veröffentlichen von Ereignissen des Typs Object in RRS lost (Objekt in RRS verloren) in Ihrem Amazon SNS-Thema.

Nachdem Sie die Benachrichtigungskonfiguration gespeichert haben, veröffentlicht Amazon S3 eine Testmeldung, die Sie per E-Mail erhalten.

Anweisungen finden Sie unter Aktivieren und Konfigurieren von Ereignis-Benachrichtigungen mit der Amazon-S3-Konsole.

Option B: Aktivieren von Benachrichtigungen für einen Bucket mithilfe der AWS SDKs

.NET

Das folgende C#-Codebeispiel zeigt ein vollständiges Listing, das einem Bucket eine Benachrichtigungskonfiguration hinzufügt. Sie müssen den Code aktualisieren und Ihren Bucket-Namen und den ARN des SNS-Themas angeben. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
```

```
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class EnableNotificationsTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string snsTopic = "*** SNS topic ARN ***";
        private const string sqsQueue = "*** SQS topic ARN ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            EnableNotificationAsync().Wait();
        }

        static async Task EnableNotificationAsync()
        {
            try
            {
               PutBucketNotificationRequest request = new
 PutBucketNotificationRequest
                {
                    BucketName = bucketName
                };

                TopicConfiguration c = new TopicConfiguration
                {
                    Events = new List<EventType> { EventType.ObjectCreatedCopy },
                    Topic = snsTopic
                };
                request.TopicConfigurations = new List<TopicConfiguration>();
                request.TopicConfigurations.Add(c);
                request.QueueConfigurations = new List<QueueConfiguration>();
                request.QueueConfigurations.Add(new QueueConfiguration()
                {
                    Events = new List<EventType> { EventType.ObjectCreatedPut },
                    Queue = sqsQueue
                });
```

```
                PutBucketNotificationResponse response = await
 client.PutBucketNotificationAsync(request);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' ",
 e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown error encountered on server.
 Message:'{0}' ", e.Message);
            }
        }
    }
}
```

Java

Die folgende Beispiel veranschaulicht, wie Sie einem Bucket eine Benachrichtigungskonfiguration hinzufügen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.EnumSet;

public class EnableNotificationOnABucket {

    public static void main(String[] args) throws IOException {
        String bucketName = "*** Bucket name ***";
        Regions clientRegion = Regions.DEFAULT_REGION;
        String snsTopicARN = "*** SNS Topic ARN ***";
        String sqsQueueARN = "*** SQS Queue ARN ***";
```

```
        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();
            BucketNotificationConfiguration notificationConfiguration = new
 BucketNotificationConfiguration();

            // Add an SNS topic notification.
            notificationConfiguration.addConfiguration("snsTopicConfig",
                    new TopicConfiguration(snsTopicARN,
 EnumSet.of(S3Event.ObjectCreated)));

            // Add an SQS queue notification.
            notificationConfiguration.addConfiguration("sqsQueueConfig",
                    new QueueConfiguration(sqsQueueARN,
 EnumSet.of(S3Event.ObjectCreated)));

            // Create the notification configuration request and set the bucket
 notification
            // configuration.
            SetBucketNotificationConfigurationRequest request = new
 SetBucketNotificationConfigurationRequest(
                    bucketName, notificationConfiguration);
            s3Client.setBucketNotificationConfiguration(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Schritt 4: Testen der Einrichtung

Jetzt können Sie die Einrichtung testen, indem Sie ein Objekt in Ihren Bucket hochladen und die Ereignis-Benachrichtigung in der Amazon-SQS-Konsole überprüfen. Anweisungen finden Sie

unter Empfangen einer Nachricht im Abschnitt "Erste Schritte" im Amazon Simple Queue Service-Entwicklerhandbuch.

Konfigurieren von Ereignis-Benachrichtigungen mithilfe der Namensfilterung

Bei der Konfiguration einer Amazon-S3-Ereignis-Benachrichtigung müssen Sie angeben, welche unterstützten Amazon-S3-Ereignistypen das Senden der Benachrichtigung durch Amazon S3 veranlassen. Wenn ein Ereignistyp, den Sie nicht angegeben haben, in Ihrem S3-Bucket auftritt, sendet Amazon S3 die Benachrichtigung nicht.

Sie können Benachrichtigungen konfigurieren, die nach dem Präfix und dem Suffix des Objektschlüsselnamens gefiltert werden. Sie können beispielsweise eine Konfiguration so einrichten, dass Sie nur eine Benachrichtigung erhalten, wenn einem Bucket Bilddateien mit der Erweiterung „`.jpg`“ hinzugefügt werden. Oder Sie können eine Konfiguration haben, die eine Benachrichtigung an ein Amazon SNS-Thema sendet, wenn dem Bucket ein Objekt mit dem Präfix „`images/`“ hinzugefügt wird, während Benachrichtigungen für Objekte mit dem Präfix „`logs/`“ im selben Bucket an eine - AWS Lambda Funktion übermittelt werden.

> **Note**
>
> Ein Platzhalterzeichen (*) kann in Filtern nicht als Präfix oder Suffix verwendet werden. Wenn Ihr Präfix oder Suffix ein Leerzeichen enthält, müssen Sie es durch das Zeichen „+“ ersetzen. Wenn Sie andere Sonderzeichen in dem Wert des Präfixes oder Suffixes verwenden, müssen Sie diese im URL-kodierten (Prozent-kodierten) Format angeben. Eine vollständige Liste der Sonderzeichen, die in ein URL-kodiertes Format umgewandelt werden müssen, wenn sie in einem Präfix oder Suffix für Ereignisbenachrichtigungen verwendet werden, finden Sie unter Sichere Zeichen.

Sie können Benachrichtigungskonfigurationen einrichten, die die Filterung von Objektschlüsselnamen in der Amazon-S3-Konsole verwenden. Sie können dies tun, indem Sie Amazon S3-APIs über die - AWS SDKs oder die REST-APIs direkt verwenden. Informationen zur Verwendung der Konsolen-Benutzeroberfläche zum Festlegen einer Benachrichtigungskonfiguration für einen Bucket finden Sie unter Aktivieren und Konfigurieren von Ereignis-Benachrichtigungen mit der Amazon-S3-Konsole.

Amazon S3 speichert die Benachrichtigungskonfiguration als XML in der einem Bucket zugeordneten notification-Subressource, wie in Verwenden von Amazon SQS, Amazon SNS und Lambda beschrieben. Sie können die `Filter` XML-Struktur verwenden, um Regeln für die nach dem Präfix

oder dem Suffix eines Objektschlüsselnamens gefilterten Benachrichtigungen zu definieren. Weitere Informationen zu der `Filter`-XML-Struktur finden Sie unter PUT-Bucket-Benachrichtigung in der API-Referenz zu Amazon Simple Storage Service.

Benachrichtigungskonfigurationen, die `Filter` verwenden, können keine Filterregeln mit überlappenden Präfixen, überlappenden Suffixen oder Präfix- und Suffix-Überlappung definieren. Die folgenden Abschnitte enthalten Beispiele für gültige Benachrichtigungskonfigurationen mit Objektschlüssel-Namensfilterung. Sie enthalten auch Beispiele für Benachrichtigungskonfigurationen, die wegen der Überlappung von Präfix und Suffix ungültig sind.

Themen

Beispiele für gültige Benachrichtigungskonfigurationen mit Filterung nach dem Objektschlüsselnamen

Die folgende Benachrichtigungskonfiguration enthält eine Warteschlangekonfiguration, die eine Amazon SQS-Warteschlange für Amazon S3 identifiziert, in der Ereignisse des Typs `s3:ObjectCreated:Put` gespeichert werden sollen. Die Ereignisse werden veröffentlicht, wenn ein Objekt mit dem Präfix `images/` und dem Suffix `jpg` mit PUT in einen Bucket geschrieben wird.

```
<NotificationConfiguration>
  <QueueConfiguration>
      <Id>1</Id>
      <Filter>
          <S3Key>
              <FilterRule>
                  <Name>prefix</Name>
                  <Value>images/</Value>
              </FilterRule>
              <FilterRule>
                  <Name>suffix</Name>
                  <Value>jpg</Value>
              </FilterRule>
          </S3Key>
      </Filter>
      <Queue>arn:aws:sqs:us-west-2:444455556666:s3notificationqueue</Queue>
      <Event>s3:ObjectCreated:Put</Event>
  </QueueConfiguration>
```

```
</NotificationConfiguration>
```

Die folgende Benachrichtigungskonfigurationen hat mehrere nicht überlappende Präfixe. Die Konfiguration definiert, dass Benachrichtigungen für PUT-Anforderungen im Ordner `images/` in queue-A geschrieben werden, während Benachrichtigungen für PUT-Anforderungen im Ordner `logs/` in queue-B geschrieben werden.

```
<NotificationConfiguration>
  <QueueConfiguration>
     <Id>1</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images/</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <Queue>arn:aws:sqs:us-west-2:444455556666:sqs-queue-A</Queue>
     <Event>s3:ObjectCreated:Put</Event>
  </QueueConfiguration>
  <QueueConfiguration>
     <Id>2</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>logs/</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <Queue>arn:aws:sqs:us-west-2:444455556666:sqs-queue-B</Queue>
     <Event>s3:ObjectCreated:Put</Event>
  </QueueConfiguration>
</NotificationConfiguration>
```

Die folgende Benachrichtigungskonfigurationen hat mehrere nicht überlappende Suffixe. Die Konfiguration definiert, dass alle dem Bucket neu hinzugefügten `.jpg`-Bilder über Lambda cloud-function-A und alle neu hinzugefügten `.png`-Bilder über cloud-function-B verarbeitet werden. Die `.png` und `.jpg` Suffixe überschneiden sich nicht, obwohl sie den gleichen Endbuchstaben haben. Wenn eine bestimmte Zeichenfolge mit beiden Suffixen enden kann, werden die beiden Suffixe als

überlappend betrachtet. Da eine Zeichenfolge nicht mit `.png` und `.jpg` gleichzeitig enden kann, sind die Suffixe in der Beispielkonfiguration keine überlappenden Suffixe.

```
<NotificationConfiguration>
  <CloudFunctionConfiguration>
     <Id>1</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.jpg</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-A</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
  <CloudFunctionConfiguration>
     <Id>2</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.png</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-B</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
</NotificationConfiguration>
```

Ihre Benachrichtigungskonfigurationen, die `Filter` verwenden, können keine Filterregeln mit überlappenden Präfixen für dieselben Ereignistypen definieren. Sie können dies nur tun, wenn die sich überlappenden Präfixe, die mit Suffixen verwendet werden, die sich nicht überlappen. Die folgende Beispielkonfiguration zeigt, wie Objekte, die mit einem gemeinsamen Präfix, aber nicht überlappenden Suffixen erstellt werden, an unterschiedliche Ziele geliefert werden können.

```
<NotificationConfiguration>
  <CloudFunctionConfiguration>
```

```
     <Id>1</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images</Value>
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.jpg</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-A</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
  <CloudFunctionConfiguration>
     <Id>2</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images</Value>
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.png</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-B</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
</NotificationConfiguration>
```

Beispiele für Benachrichtigungskonfigurationen mit ungültiger Präfix- und Suffix-Überlappung

Ihre Benachrichtigungskonfigurationen, die `Filter` verwenden, können größtenteils keine Filterregeln mit überlappenden Präfixen, überlappenden Suffixen oder überlappenden Präfix- und Suffix-Kombinationen für dieselben Ereignistypen definieren. Überlappende Präfixe sind möglich,

solange sich die Suffixe nicht überlappen. Ein Beispiel finden Sie unter Konfigurieren von Ereignis-Benachrichtigungen mithilfe der Namensfilterung.

Sie können überlappende Objektschlüsselnamensfilter mit unterschiedlichen Ereignistypen verwenden. Beispielsweise könnten sie eine Benachrichtigungskonfiguration erstellen, die das Präfix `image/` für den Ereignistyp `ObjectCreated:Put` und das Präfix `image/` für den Ereignistyp `ObjectRemoved:*` verwendet.

Sie erhalten einen Fehler, wenn Sie versuchen, eine Benachrichtigungskonfiguration mit ungültigen, sich überschneidenden Namensfiltern für dieselben Ereignistypen zu speichern, wenn Sie die Amazon-S3-Konsole oder -API verwenden. Dieser Abschnitt zeigt Beispiele für Benachrichtigungskonfigurationen, die aufgrund überlappender Namensfilter ungültig sind.

Es wird angenommen, dass eine vorhandene Benachrichtigungskonfigurationsregel ein Standardpräfix und -suffix hat, die mit allen anderen Präfixen bzw. Suffixen übereinstimmen. Die folgende Benachrichtigungskonfiguration ist ungültig, weil sie überlappende Präfixe hat. Insbesondere überlappt sich das Root-Präfix mit allen anderen Präfixen. Das gleiche gilt, wenn Sie in diesem Beispiel ein Suffix anstelle eines Präfixes verwenden. Das Root-Suffix überlappt mit allen anderen Suffixen.

```
<NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-notification-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-notification-two</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
                     <Value>images</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
</NotificationConfiguration>
```

Die folgende Benachrichtigungskonfiguration ist ungültig, weil sie überlappende Suffixe hat. Wenn eine bestimmte Zeichenfolge mit beiden Suffixen enden kann, werden die beiden Suffixe als

überlappend betrachtet. Eine Zeichenfolge kann mit `jpg` und `pg` enden. Die Suffixe überschneiden sich also. Gleiches gilt für Präfixe. Wenn eine bestimmte Zeichenfolge mit beiden Präfixen beginnen kann, werden die beiden Präfixe als überlappend betrachtet.

```
 <NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>jpg</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-two</Topic>
         <Event>s3:ObjectCreated:Put</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>pg</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
</NotificationConfiguration
```

Die folgende Benachrichtigungskonfiguration ist ungültig, weil sie überlappende Präfixe und Suffixe hat.

```
<NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
```

```
                    <Value>images</Value>
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>jpg</Value>
                </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-two</Topic>
         <Event>s3:ObjectCreated:Put</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>jpg</Value>
                 </FilterRule>
             </S3Key>
         </Filter>
    </TopicConfiguration>
</NotificationConfiguration>
```

Struktur von Ereignismeldungen

Die Benachrichtigung, die Amazon S3 sendet, um ein Ereignis zu veröffentlichen, verwendet das JSON-Format.

Eine allgemeine Übersicht und Anweisungen zum Konfigurieren von Ereignisbenachrichtigungen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.

Dieses Beispiel veranschaulicht Version 2.2 der JSON-Struktur der Ereignisbenachrichtigung. Amazon S3 verwendet Versionen 2.1 und 2.2 und 2.3 dieser Ereignisstruktur. Amazon S3 verwendet Version 2.2 für regionsübergreifende Replikationsereignisbenachrichtigungen. Es verwendet Version 2.3 für S3 Lebenszyklus, S3 Intelligent-Tiering, Objekt-ACL, Objekt-Markierung und Objektwiederherstellungs-Löschereignisse. Diese Versionen enthalten zusätzliche Informationen, die für diese Vorgänge spezifisch sind. Die Versionen 2.2 und 2.3 sind ansonsten mit Version 2.1 kompatibel, die Amazon S3 derzeit für alle anderen Ereignisbenachrichtigungstypen verwendet.

```
{
   "Records":[
      {
```

```
        "eventVersion":"2.2",
        "eventSource":"aws:s3",
        "awsRegion":"us-west-2",
        "eventTime":"The time, in ISO-8601 format, for example,
1970-01-01T00:00:00.000Z, when Amazon S3 finished processing the request",
        "eventName":"event-type",
        "userIdentity":{
           "principalId":"Amazon-customer-ID-of-the-user-who-caused-the-event"
        },
        "requestParameters":{
           "sourceIPAddress":"ip-address-where-request-came-from"
        },
        "responseElements":{
           "x-amz-request-id":"Amazon S3 generated request ID",
           "x-amz-id-2":"Amazon S3 host that processed the request"
        },
        "s3":{
           "s3SchemaVersion":"1.0",
           "configurationId":"ID found in the bucket notification configuration",
           "bucket":{
              "name":"bucket-name",
              "ownerIdentity":{
                 "principalId":"Amazon-customer-ID-of-the-bucket-owner"
              },
              "arn":"bucket-ARN"
           },
           "object":{
              "key":"object-key",
              "size":"object-size in bytes",
              "eTag":"object eTag",
              "versionId":"object version if bucket is versioning-enabled, otherwise
null",
              "sequencer": "a string representation of a hexadecimal value used to
determine event sequence, only used with PUTs and DELETEs"
           }
        },
        "glacierEventData": {
           "restoreEventData": {
              "lifecycleRestorationExpiryTime": "The time, in ISO-8601 format, for
example, 1970-01-01T00:00:00.000Z, of Restore Expiry",
              "lifecycleRestoreStorageClass": "Source storage class for restore"
           }
        }
     }
```

```
    ]
}
```

Beachten Sie im Zusammenhang mit der Ereignistachrichtenstruktur Folgendes:

- Der `eventVersion`-Schlüsselwert enthält eine Haupt- und eine Nebenversion im Format `<major>.<minor>`.

  Die Hauptversion wird erhöht, wenn Amazon S3 eine Änderung an der Ereignisstruktur vornimmt, die nicht abwärtskompatibel ist. Dies beinhaltet das Entfernen eines JSON-Feldes, das bereits vorhanden ist, oder das Ändern, wie die Inhalte eines Feldes dargestellt werden (Beispiel: ein Datumsformat).

  Die Nebenversion wird erhöht, wenn Amazon S3 der Ereignisstruktur neue Felder hinzufügt. Dies kann auftreten, wenn neue Informationen für einige oder alle vorhandenen Ereignisse bereitgestellt werden. Dies kann auch geschehen, wenn neue Informationen nur zu neu eingeführten Ereignistypen bereitgestellt werden. Anwendungen sollten neue Felder ignorieren, um weiter aufwärtskompatibel mit neuen Nebenversionen der Ereignisstruktur zu sein.

  Falls neue Ereignistypen eingeführt werden, aber die Struktur des Ereignisses anderweitig unmodifiziert ist, ändert sich die Ereignisversion nicht.

  Um sicherzustellen, dass Ihre Anwendungen die Ereignisstruktur ordnungsgemäß analysieren können, empfehlen wir, dass Sie einen Vergleich mit der Hauptversionsnummer durchführen. Um sicherzustellen, dass die Felder, die von Ihrer Anwendung erwartet werden, vorhanden sind, empfehlen wir auch, einen greater-than-or-equal-zu-Vergleich mit der Nebenversion durchzuführen.
- Die `eventName` verweist auf die Liste der Typen der Ereignisbenachrichtigung, enthält aber nicht die `s3:`-Präfix.
- Der `responseElements` Schlüsselwert ist nützlich, wenn Sie eine Anforderung verfolgen möchten, indem Sie mit fortfahren AWS Support. `x-amz-request-id` und `x-amz-id-2` helfen Amazon S3, eine einzelne Anfrage nachzuverfolgen. Diese Werte sind dieselben, die Amazon S3 in der Antwort auf die Anforderung zurückgibt, die die Ereignisse initiiert. Auf diese Weise können sie verwendet werden, um das Ereignis der Anfrage zuzuordnen.
- Der `s3`-Schlüssel bietet Informationen über den Bucket und das Objekt, die an dem Ereignis beteiligt sind. Der Wert des Objektschlüsselnamens ist URL-kodiert. Beispielsweise wird "red flower.jpg" zu "red+flower.jpg" (Amazon S3 gibt als Inhaltstyp in der Antwort "`application/x-www-form-urlencoded`" zurück).

- Der `sequencer`-Schlüssel bietet eine Möglichkeit, die Reihenfolge von Ereignissen zu bestimmen. Ereignis-Benachrichtigungen kommen nicht garantiert in der Reihenfolge an, in der die Ereignisse aufgetreten sind. Benachrichtigungen von Ereignissen, die Objekte erstellen (PUT) und Objekte löschen, enthalten jedoch ein `sequencer`. Es kann verwendet werden, um die Reihenfolge der Ereignisse für einen bestimmten Objektschlüssel zu bestimmen.

  Wenn Sie die `sequencer`-Zeichenfolgen von zwei Ereignisbenachrichtigungen für denselben Objektschlüssel vergleichen, ist die Ereignisbenachrichtigung mit dem größeren hexadezimalen Wert von `sequencer` das später aufgetretene Ereignis. Wenn Sie Ereignisbenachrichtigungen verwenden, um eine separate Datenbank oder einen separaten Index Ihrer Amazon-S3-Objekte zu verwalten, empfehlen wir Ihnen, die `sequencer`-Werte während der Verarbeitung jeder Ereignisbenachrichtigung zu vergleichen und zu speichern.

  Beachten Sie Folgendes:

  - Sie können `sequencer` nicht verwenden, um die Reihenfolge von Ereignissen für unterschiedliche Objektschlüssel zu bestimmen.
  - Die Sequenzer können unterschiedliche Längen haben. Um diese Werte zu vergleichen, füllen Sie zuerst den kürzeren Wert links mit Nullen auf und führen dann einen alphabetischen Vergleich durch.
- Der `glacierEventData`-Schlüssel ist nur für `s3:ObjectRestore:Completed`-Ereignisse sichtbar.
- Der `restoreEventData`-Schlüssel enthält Attribute, die sich auf Ihre Wiederherstellungsanfrage beziehen.
- Der Schlüssel `replicationEventData` ist nur für Replikationsereignisse sichtbar.
- Der `intelligentTieringEventData`-Schlüssel ist nur für Ereignisse von S3 Intelligent Tiering sichtbar.
- Der `lifecycleEventData`-Schlüssel ist nur für S3-Lebenszyklus-Übergangsereignisse sichtbar.

Beispielnachrichten

Im Folgenden finden Sie Beispiele für Amazon-S3-Ereignis-Benachrichtigungen.

Amazon-S3-Testnachricht

Nachdem Sie eine Ereignis-Benachrichtigung für einen Bucket konfigurieren, sendet Amazon S3 die folgende Testnachricht.

```
{
   "Service":"Amazon S3",
   "Event":"s3:TestEvent",
   "Time":"2014-10-13T15:57:02.089Z",
   "Bucket":"bucketname",
   "RequestId":"5582815E1AEA5ADF",
   "HostId":"8cLeGAmw098X5cv4Zkwcmo8vvZa3eH3eKxsPzbB9wrR+YstdA6Knx4Ip8EXAMPLE"
}
```

Beispielnachricht, wenn ein Objekt mit einer PUT-Anforderung erstellt wird

Die folgende Nachricht ist ein Beispiel für eine Nachricht, die Amazon S3 zum Veröffentlichen eines `s3:ObjectCreated:Put`-Ereignisses sendet.

```
{
   "Records":[
      {
         "eventVersion":"2.1",
         "eventSource":"aws:s3",
         "awsRegion":"us-west-2",
         "eventTime":"1970-01-01T00:00:00.000Z",
         "eventName":"ObjectCreated:Put",
         "userIdentity":{
            "principalId":"AIDAJDPLRKLG7UEXAMPLE"
         },
         "requestParameters":{
            "sourceIPAddress":"127.0.0.1"
         },
         "responseElements":{
            "x-amz-request-id":"C3D13FE58DE4C810",
            "x-amz-id-2":"FMyUVURIY8/IgAtTv8xRjskZQpcIZ9KG4V5Wp6S7S/
JRWeUWerMUE5JgHvANOjpD"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"testConfigRule",
            "bucket":{
               "name":"mybucket",
               "ownerIdentity":{
                  "principalId":"A3NL1KOZZKExample"
               },
               "arn":"arn:aws:s3:::mybucket"
            },
```

```
            "object":{
               "key":"HappyFace.jpg",
               "size":1024,
               "eTag":"d41d8cd98f00b204e9800998ecf8427e",
               "versionId":"096fKKXTRTtl3on89fVO.nfljtsv6qko",
               "sequencer":"0055AED6DCD90281E5"
            }
         }
      }
   ]
}
```

Eine Definition der einzelnen IAM-Identifikationspräfixe (z. B. AIDA, AROA, AGPA) finden Sie unter IAM-IDs im IAM-Benutzerhandbuch.

Verwenden von EventBridge

Amazon S3 kann Ereignisse an Amazon senden, EventBridge wenn bestimmte Ereignisse in Ihrem Bucket auftreten. Im Gegensatz zu anderen Zielen müssen Sie nicht auswählen, welche Ereignistypen Sie liefern möchten. Nachdem aktiviert EventBridge wurde, werden alle folgenden Ereignisse an gesendet EventBridge. Sie können EventBridge Regeln verwenden, um Ereignisse an zusätzliche Ziele weiterzuleiten. Im Folgenden werden die Ereignisse aufgeführt, die Amazon S3 an sendet EventBridge.

Ereignistyp	Beschreibung
Objekt erstellt	Ein Objekt wurde erstellt. Das Feld Grund in der Struktur der Ereignisnachricht gibt an, welche S3-API zum Erstellen des Objekts verwendet wurde: PutObject, POST ObjectCopyObject , oder CompleteMultipartUpload.
Objekt gelöscht (DeleteObject) Objekt gelöscht (Lebenszy klusablauf)	Ein Objekt wurde gelöscht. Wenn ein Objekt mithilfe eines S3-API-Aufrufs gelöscht wird, wird das Ursachenfeld auf gesetzt DeleteObject. Wenn ein Objekt durch eine S3-Lebenszyklus-Ablaufregel gelöscht wird, wird das Grundfeld auf Lebenszyklus-Ablau

Ereignistyp	Beschreibung
	f gesetzt. Weitere Informationen finden Sie unter Auslaufende Objekte. Wenn ein nicht versioniertes Objekt gelöscht wird oder ein versioniertes Objekt dauerhaft gelöscht wird, wird das Feld für den Löschtyp auf Dauerhaft gelöscht gesetzt. Wenn eine Löschmarkierung für ein versioniertes Objekt erstellt wird, wird das Feld für den Löschtyp auf Löschmarkierung erstellt gesetzt. Weitere Informationen finden Sie unter Löschen von Objekten aus einem versioning-fähigen Bucket.
Objektwiederherstellung eingeleitet	Eine Objektwiederherstellung wurde von der Speicherklasse S3 Glacier oder S3 Glacier Deep Archive oder von der Ebene S3 Intelligent-Tiering Archive Access oder Deep Archive Access initiiert. Weitere Informationen finden Sie unter Arbeiten mit archivierten Objekten.
Objektwiederherstellung abgeschlossen	Eine Objektwiederherstellung wurde abgeschlossen.
Wiederherstellen von Objekten ist abgelaufen	Die temporäre Kopie eines Objekts, das aus S3 Glacier oder S3 Glacier Deep Archive wiederhergestellt wurde, abgelaufen und wurde gelöscht.
Objektspeicherklasse geändert	Ein Objekt wurde auf eine andere Speicherklasse umgestellt. Weitere Informationen finden Sie unter Übergang von Objekten mit Amazon-S3-Lebenszyklus.
Objektzugriffsebene wurde geändert	Ein Objekt wurde auf die Stufen S3 Intelligent-Tiering Archive Access oder Deep Archive Access umgestellt. Weitere Informationen finden Sie unter Amazon S3 Intelligent Tiering.

Ereignistyp	Beschreibung
ACL-Objekt aktualisiert	Die Zugriffssteuerungsliste (ACL) eines Objekts wurde mithilfe von PutObjectACL festgelegt. Ein Ereignis wird nicht generiert, wenn eine Anforderung keine Änderung an der ACL eines Objekts zur Folge hat. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.
Objekt-Tags hinzugefügt	Einem Objekt wurde mit ein Satz von Tags hinzugefügt PutObjectTagging. Weitere Informationen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.
Gelöschte Objekt-Tags	Alle Tags wurden mit aus einem Objekt entfernt DeleteObjectTagging. Weitere Informationen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.

> **Note**
>
> Weitere Informationen zur Zuordnung von Amazon S3-Ereignistypen zu EventBridge Ereignistypen finden Sie unter Amazon EventBridge -Zuweisung und Fehlerbehebung.

Sie können Amazon S3-Ereignisbenachrichtigungen mit verwenden, EventBridge um Regeln zu schreiben, die Aktionen ausführen, wenn ein Ereignis in Ihrem Bucket eintritt. Sie können sich beispielsweise eine Benachrichtigung senden lassen. Weitere Informationen finden Sie unter Was ist EventBridge im Amazon- EventBridge Benutzerhandbuch.

Weitere Informationen zu Preisen finden Sie unter Amazon- EventBridge Preise.

Themen

- Amazon- EventBridge Berechtigungen
- Aktivieren von Amazon EventBridge
- EventBridge Struktur der Ereignisnachricht
- Amazon EventBridge -Zuweisung und Fehlerbehebung

Amazon- EventBridge Berechtigungen

Amazon S3 benötigt keine zusätzlichen Berechtigungen, um Ereignisse an Amazon zu übermitteln EventBridge.

Aktivieren von Amazon EventBridge

Sie können Amazon EventBridge über die S3-Konsole, AWS Command Line Interface (AWS CLI) oder die Amazon S3-REST-API aktivieren.

Verwenden der S3-Konsole

So aktivieren Sie die EventBridge Ereigniszustellung in der S3-Konsole.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie Ereignisse aktivieren möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Navigieren Sie zum Abschnitt Ereignisbenachrichtigungen und suchen Sie den Unterabschnitt Amazon EventBridge. Wählen Sie Bearbeiten aus.
5. Wählen Sie unter Benachrichtigungen an Amazon EventBridge für alle Ereignisse in diesem Bucket senden die Option Ein aus.

> **Note**
>
> Nachdem Sie aktiviert haben EventBridge, dauert es etwa fünf Minuten, bis die Änderungen wirksam werden.

Verwenden der AWS CLI

Im folgenden Beispiel wird eine Bucket-Benachrichtigungskonfiguration für einen Bucket `DOC-EXAMPLE-BUCKET1` mit EventBridge aktiviertem Amazon erstellt.

```
aws s3api put-bucket-notification-configuration --bucket DOC-EXAMPLE-BUCKET1 --
notification-configuration='{ "EventBridgeConfiguration": {} }'
```

Verwenden der REST-API

Sie können Amazon programmgesteuert EventBridge für einen Bucket aktivieren, indem Sie die Amazon S3-REST-API aufrufen. Weitere Informationen finden Sie unter PutBucketNotificationConfiguration in der API-Referenz zu Amazon Simple Storage Service.

Das folgende Beispiel zeigt das XML, das zum Erstellen einer Bucket-Benachrichtigungskonfiguration mit EventBridge aktiviertem Amazon verwendet wird.

```
<NotificationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <EventBridgeConfiguration>
  </EventBridgeConfiguration>
</NotificationConfiguration>
```

Erstellen von EventBridge Regeln

Nach der Aktivierung können Sie Amazon- EventBridge Regeln für bestimmte Aufgaben erstellen. Beispielsweise können Sie E-Mail-Benachrichtigungen senden, wenn ein Objekt erstellt wird. Ein vollständiges Tutorial finden Sie unter Tutorial: Senden einer Benachrichtigung, wenn ein Amazon S3-Objekt erstellt wird im Amazon- EventBridge Benutzerhandbuch.

EventBridge Struktur der Ereignisnachricht

Die Benachrichtigung, die Amazon S3 sendet, um ein Ereignis zu veröffentlichen, verwendet das JSON-Format. Wenn Amazon S3 ein Ereignis an Amazon sendet EventBridge, sind die folgenden Felder vorhanden.

- Ausführung – Derzeit 0 (Null) für alle Ereignisse.
- id – Eine UUID der Version 4, die für jedes Ereignis generiert wurde.
- detail-type – Die Art des Ereignisses, das gesendet wird. Eine Liste der Ereignistypen finden Sie unter Verwenden von EventBridge.
- Quelle – Gibt den Service an, aus dem das Ereignis stammt.
- Konto– Die 12-stellige AWS-Konto -ID des Bucket-Eigentümers.
- Zeit – Die Zeit, zu der das Ereignis aufgetreten ist.
- Region – Identifiziert den AWS-Region des Buckets.
- Ressource – Ein JSON-Array, das den Amazon-Ressourcennamen (ARN) des Buckets enthält.
- Detail – Ein JSON-Objekt, das Informationen zum Ereignis enthält. Weitere Informationen dazu, was in diesem Feld enthalten sein kann, finden Sie unter Detailfeld für Ereignismeldung.

Beispiele für Ereignismeldungen

Im Folgenden finden Sie Beispiele für einige der Amazon S3-Ereignisbenachrichtigungen, die an Amazon gesendet werden können EventBridge.

Objekt erstellt

```
{
  "version": "0",
  "id": "17793124-05d4-b198-2fde-7ededc63b103",
  "detail-type": "Object Created",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "size": 5,
      "etag": "b1946ac92492d2347c6235b4d2611184",
      "version-id": "IYV3p45BT0ac8hjHg1houSdS1a.Mro8e",
      "sequencer": "617f08299329d189"
    },
    "request-id": "N4N7GDK58NMKJ12R",
    "requester": "123456789012",
    "source-ip-address": "1.2.3.4",
    "reason": "PutObject"
  }
}
```

Objekt gelöscht (mit DeleteObject)

```
{
  "version": "0",
  "id": "2ee9cc15-d022-99ea-1fb8-1b1bac4850f9",
```

```
  "detail-type": "Object Deleted",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "etag": "d41d8cd98f00b204e9800998ecf8427e",
      "version-id": "1QW9g1Z99LUNbvaaYVpW9xDlOLU.qxgF",
      "sequencer": "617f0837b476e463"
    },
    "request-id": "0BH729840619AG5K",
    "requester": "123456789012",
    "source-ip-address": "1.2.3.4",
    "reason": "DeleteObject",
    "deletion-type": "Delete Marker Created"
  }
}
```

Objekt wurde gelöscht (unter Verwendung des Lebenszyklusablaufs)

```
{
  "version": "0",
  "id": "ad1de317-e409-eba2-9552-30113f8d88e3",
  "detail-type": "Object Deleted",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
```

```
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "etag": "d41d8cd98f00b204e9800998ecf8427e",
      "version-id": "mtB0cV.jejK63XkRNceanNMC.qXPWLeK",
      "sequencer": "617b398000000000"
    },
    "request-id": "20EB74C14654DC47",
    "requester": "s3.amazonaws.com",
    "reason": "Lifecycle Expiration",
    "deletion-type": "Delete Marker Created"
  }
}
```

Objektwiederherstellung abgeschlossen

```
{
  "version": "0",
  "id": "6924de0d-13e2-6bbf-c0c1-b903b753565e",
  "detail-type": "Object Restore Completed",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "size": 5,
      "etag": "b1946ac92492d2347c6235b4d2611184",
      "version-id": "KKsjUC1.6gIjqtvhfg5AdMI0eCePIiT3"
    },
    "request-id": "189F19CB7FB1B6A4",
    "requester": "s3.amazonaws.com",
```

```
    "restore-expiry-time": "2021-11-13T00:00:00Z",
    "source-storage-class": "GLACIER"
  }
}
```

Detailfeld für Ereignismeldung

Das Detailfeld enthält ein JSON-Objekt mit Informationen über das Ereignis. Die folgenden Felder können im Detailfeld vorhanden sein.

- Ausführung – Derzeit 0 (Null) für alle Ereignisse.
- Bucket – Informationen über den Amazon-S3-Bucket, der an der Veranstaltung beteiligt ist.
- Objekt – Informationen über das an dem Ereignis beteiligte Amazon-S3-Objekt.
- request-id – Anforderungs-ID in S3 Antwort.
- Anforderer – AWS-Konto ID oder AWS Service-Prinzipal des Anforderers.
- source-ip-address – Quell-IP-Adresse der S3-Anforderung. Nur vorhanden für Ereignisse, die durch eine S3-Anfrage ausgelöst werden.
- Grund – Für Object Created Events die S3-API, die zum Erstellen des Objekts verwendet wurde: PutObject, POST Object , CopyObjectoder CompleteMultipartUpload. Bei Ereignissen mit Objektlöschung wird dies auf gesetzt, DeleteObject wenn ein Objekt durch einen S3-API-Aufruf gelöscht wird, oder auf Lebenszyklusablauf, wenn ein Objekt durch eine S3-Lebenszyklusablaufregel gelöscht wird. Weitere Informationen finden Sie unter Auslaufende Objekte.
- deletion-type – Wenn ein nicht versioniertes Objekt gelöscht wird oder ein versioniertes Objekt dauerhaft gelöscht wird, wird bei Object Deleted-Ereignissen dies auf Permanently Deleted festgelegt. Wenn ein Löschmarker für ein versioniertes Objekt erstellt wird, wird dieser auf Löschmarkierung erstellt gesetzt. Weitere Informationen finden Sie unter Löschen von Objekten aus einem versioning-fähigen Bucket.
- restore-expiry-time – Bei Ereignissen mit abgeschlossener Objektwiederherstellung der Zeitpunkt, zu dem die temporäre Kopie des Objekts aus S3 gelöscht wird. Weitere Informationen finden Sie unter Arbeiten mit archivierten Objekten.
- source-storage-class – Bei den Ereignissen Object Restore Initiated und Object Restore Completed die Speicherklasse des wiederherzustellenden Objekts. Weitere Informationen finden Sie unter Arbeiten mit archivierten Objekten.

- destination-storage-class – Bei geänderten Ereignissen der Objektspeicherklasse die neue Speicherklasse des Objekts. Weitere Informationen finden Sie unter Übergang von Objekten mit Amazon-S3-Lebenszyklus.
- destination-access-tier – Bei geänderten Ereignissen der Objektzugriffsebene die neue Zugriffsebene des Objekts. Weitere Informationen finden Sie unter Amazon S3 Intelligent Tiering.

Amazon EventBridge -Zuweisung und Fehlerbehebung

In der folgenden Tabelle wird beschrieben, wie Amazon S3-Ereignistypen Amazon- EventBridge Ereignistypen zugeordnet werden.

S3-Ereignistyp	Amazon- EventBridge Detailtyp
ObjectCreated:Put ObjectCreated:Post ObjectCreated:Kopie ObjectCreated:CompleteMultipartUpload	Objekt erstellt
ObjectRemoved:Löschen ObjectRemoved:DeleteMarkerCreated LifecycleExpiration:Löschen LifecycleExpiration:DeleteMarkerCreated	Objekt gelöscht
ObjectRestore:Post	Objektwiederherstellung eingeleitet
ObjectRestore:Abgeschlossen	Objektwiederherstellung abgeschlossen
ObjectRestore:Löschen	Wiederherstellen von Objekten ist abgelaufen
LifecycleTransition	Objektspeicherklasse geändert

S3-Ereignistyp	Amazon- EventBridge Detailtyp
IntelligentTiering	Objektzugriffsebene wurde geändert
ObjectTagging: Put	Objekt-Tags hinzugefügt
ObjectTagging:Löschen	Gelöschte Objekt-Tags
ObjectAcl:Put	ACL-Objekt aktualisiert

Amazon- EventBridge Fehlerbehebung

Informationen zur Fehlerbehebung finden Sie EventBridgeunter Fehlerbehebung bei Amazon EventBridge im Amazon- EventBridge Benutzerhandbuch.

Verwenden von Analysen und Einblicken

Sie können Analytik und Erkenntnisse in Amazon S3 verwenden, um Ihre Speichernutzung zu verstehen, zu analysieren und zu optimieren. Weitere Informationen finden Sie in den folgenden Themen.

Themen

Amazon S3 analytics – Speicherklassen-Analyse

Mit der Speicherklassen-Analyse von Amazon S3 Analytics können Sie Speicherzugriffsmuster analysieren, anhand derer Sie entscheiden können, ob Sie die richtigen Daten in die richtige Speicherklasse einordnen. Diese neue analytische Funktion von Amazon S3 beobachtet Datenzugriffsmuster, anhand derer Sie entscheiden können, wann Sie STANDARD-Speicher mit weniger häufigem Zugriff in die Speicherklasse STANDARD_IA (IA für „infrequent access“ (seltener Zugriff)) überführen sollen. Weitere Informationen über Speicherklassen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.

Wenn die Speicherklassen-Analyse die seltenen Zugriffsmuster für eine gefilterte Datenmenge im Laufe der Zeit beobachtet, können Sie Ihre Lebenszykluskonfigurationen unter Verwendung der Analyseergebnisse verbessern. Sie können die Speicherklassen-Analyse so konfigurieren, dass alle Objekte in einem Bucket analysiert werden. Sie können aber auch Filter konfigurieren, um Objekte nach einem gemeinsamen Präfix zu gruppieren (d. h. für Objekte, deren Namen mit einer gemeinsamen Zeichenkette beginnen), nach Objekt-Markierungen oder sowohl nach Präfix und Markierungen. Wahrscheinlich werden Sie feststellen, dass die Filterung nach Objektgruppen die beste Methode darstellt, von der Speicherklassen-Analyse zu profitieren.

> **Important**
>
> Die Speicherklassenanalyse stellt nur Empfehlungen für die Klassen Standard bis Standard-IA bereit.

Sie können mehrere Speicherklassen-Analysefilter pro Bucket einrichten, bis zu 1000, und erhalten für jeden Filter eine separate Analyse. Durch Verwendung mehrerer Filterkonfigurationen können Sie spezifische Objektgruppen analysieren, um Ihre Lebenszykluskonfigurationen zu verbessern, die Objekte in STANDARD_IA überführen.

Die Speicherklassen-Analyse bietet Schaubilder zur Speichernutzung in der Amazon-S3-Konsole, die täglich aktualisiert werden. Sie können diese täglichen Nutzungsdaten auch in einen S3-Bucket exportieren und sie in einer Tabellenkalkulationsanwendung oder mit Business-Intelligence-Tools wie Amazon anzeigen QuickSight.

Mit der Speicherklassen-Analyse sind Kosten verbunden. Preisinformationen finden Sie unter Verwaltung und Replikation Amazon-S3-Preise.

Themen

Wie richte ich die Speicherklassen-Analyse ein?

Sie können die Speicherklassen-Analyse Einrichten, indem Sie konfigurieren, welche Objektdaten analysiert werden sollen. Sie können die Speicherklassenanalsyse für die folgenden Aufgaben konfigurieren:

- Analyse des gesamten Inhalts eines Buckets.

  Sie erhalten eine Analyse für alle Objekte im Bucket.
- Analyse von nach Präfix und Tags gruppierten Objekten.

  Sie können Filter konfigurieren, um Objekte nach einem gemeinsamen Präfix oder nach Objekt-Markierungen zu gruppieren, oder nach einer Kombination aus Präfix und Markierungen. Sie erhalten für jeden konfigurierten Filter eine separate Analyse. Sie können maximal 1000 Filterkonfigurationen pro Bucket verwenden.
- Export von Analysedaten.

  Wenn Sie die Speicherklassen-Analyse für einen Bucket oder Filter konfigurieren, können Sie die Analysedaten täglich in eine Datei exportieren lassen. Die Analyse für den Tag wird der Datei

hinzugefügt, um ein Verlaufsanalyseprotokoll für den konfigurierten Filter zu erstellen. Die Datei wird täglich an dem Ziel Ihrer Wahl aktualisiert. Wenn Sie Dateien zum Export auswählen, geben Sie einen Ziel-Bucket und ein optionales Ziel-Präfix an, wohin die Datei geschrieben werden soll.

Sie können die Amazon S3-Konsole, die REST-API oder die - AWS CLI oder - AWS SDKs verwenden, um die Speicherklassenanalyse zu konfigurieren.

- Informationen zum Konfigurieren der Analyse von Speicherklassen in der Amazon-S3-Konsole finden Sie unter Konfigurieren der Speicherklassen-Analyse.
- Um die Amazon S3-API zu verwenden, verwenden Sie die PutBucketAnalyticsConfiguration REST-API oder das Äquivalent aus der AWS CLI oder den AWS SDKs .

Wie verwende ich die Speicherklassen-Analyse?

Sie verwenden die Speicherklassen-Analyse, um Datenzugriffsmuster im Laufe der Zeit zu analysieren, um Informationen zu erhalten, die Ihnen beim Lebenszyklusmanagement Ihres STANDARD_IA-Speichers helfen. Nachdem Sie einen Filter konfiguriert haben, sehen Sie die Datenanalyse basierend auf dem Filter in der Amazon-S3-Konsole über 24 bis 48 Stunden. Die Speicherklassen-Analyse beobachtet jedoch die Zugriffsmuster einer gefilterten Datenmenge für 30 Tage oder Länger, um Informationen für eine Analyse zu sammeln, bevor ein Ergebnis zurückgegeben wird. Die Analyse wird nach dem ersten Ergebnis fortgesetzt und aktualisiert das Ergebnis, wenn sich die Zugriffsmuster ändern.

Wenn Sie einen Filter zum ersten Mal konfigurieren, dauert es möglicherweise einen Moment, um die Daten der Amazon-S3-Konsole zu analysieren.

Die Speicherklassen-Analyse beobachtet die Zugriffsmuster einer gefilterten Objekt-Datenmenge für 30 Tage oder Länger, um ausreichend viele Informationen für eine Analyse zu sammeln. Nachdem die Speicherklassen-Analyse genügend Informationen gesammelt hat, wird in der Amazon-S3-Konsole eine Meldung angezeigt, dass die Analyse abgeschlossen ist.

Bei der Durchführung der Analyse für Objekte mit seltenem Zugriff betrachtet die Speicherklassen-Analyse die gefilterte gruppierte Objektmenge basierend auf der Dauer für die sie in Amazon S3 hochgeladen waren. Die Speicherklassen-Analyse stellt fest, ob für die Altersgruppe selten ein Zugriff stattfindet, indem sie die folgenden Faktoren für die gefilterte Datenmenge betrachtet.

- Objekte in der Speicherklasse STANDARD, die größer als 128 KB sind.
- Wie viel durchschnittlichen Gesamtspeicher Sie pro Altersgruppe haben.

- Die durchschnittliche Anzahl an Bytes, die pro Altersgruppe nach außen übertragen werden (nicht die Frequenz).
- Analyseexportdaten enthalten nur Anfragen mit Daten, die für die Speicherklassen-Analyse relevant sind. Dies kann Differenzen in der Anzahl der Anfragen sowie im Hinblick auf die gesamten Bytes für Upload und Anfrage im Vergleich zu den Angaben in der Speichermetrik oder in den Ergebnissen Ihrer eigenen internen Systeme verursachen.
- Fehlgeschlagene GET- und PUT-Anfragen werden bei der Analyse nicht berücksichtigt. In den Speichermetriken sehen Sie jedoch fehlgeschlagene Anfragen.

Wie viel von meinem Speicher habe ich abgerufen?

Die Amazon-S3-Konsole zeigt in einer Grafik an, wie viel von dem Speicher in der gefilterten Datenmenge für den Beobachtungszeitraum abgerufen wurde.

Welchen Prozentsatz meines Speichers habe ich abgerufen?

Die Amazon-S3-Konsole zeigt in einer Grafik an, welcher Prozentsatz des Speichers in der gefilterten Datenmenge für den Beobachtungszeitraum abgerufen wurde.

Wie in diesem Thema bereits dargelegt, betrachtet die Speicherklassen-Analyse bei der Analyse für Objekte mit seltenem Zugriff die gefilterte Objektemenge, gruppiert nach dem Zeitpunkt, zu dem die Objekte in Amazon S3 hochgeladen wurden. Die Speicherklassen-Analyse verwendet die folgenden vordefinierten Objektaltersgruppen:

- Amazon-S3-Objekte, die weniger als 15 Tage alt sind
- Amazon S3 Objekte 15-29 Tage alt
- Amazon S3 Objekte 30-44 Tage alt
- Amazon S3 Objekte 45-59 Tage alt
- Amazon S3 Objekte 60-74 Tage alt
- Amazon S3 Objekte 75-89 Tage alt
- Amazon S3 Objekte 90-119 Tage alt
- Amazon S3 Objekte 120-149 Tage alt
- Amazon S3 Objekte 150-179 Tage alt
- Amazon S3 Objekte 180-364 Tage alt
- Amazon S3 Objekte 365-729 Tage alt

- Amazon-S3-Objekte, 730 Tage alt und älter

In der Regel benötigt sie über 30 Tage beobachtete Zugriffsmuster, um ausreichend viele Informationen für ein Analyseergebnis zu sammeln. Abhängig von dem spezifischen Zugriffsmuster Ihrer Daten könnte dies länger als 30 Tage dauern. Nachdem Sie jedoch einen Filter konfiguriert haben, sehen Sie die Datenanalyse basierend auf dem Filter in der Amazon-S3-Konsole über 24 bis 48 Stunden. Sie sehen die Analyse zum Objektzugriff auf täglicher Basis, unterteilt nach Objektaltersgruppe in der Amazon-S3-Konsole.

Wie viel von meinem Speicher wird selten abgerufen?

Die Amazon-S3-Konsole zeigt die Zugriffsmuster, die nach den vordefinierten Altersgruppen des Objekts gruppiert sind. Der angezeigte Text Frequently accessed (Häufig aufgerufen) oder Infrequently accessed (Selten aufgerufen) ist als visuelle Hilfe bei der Erstellung des Lebenszyklus gedacht.

Wie kann ich die Daten der Speicherklassen-Analyse exportieren?

Sie können festlegen, dass die Speicherklassen-Analyse Analyseberichte in einer flachen .csv-Datei (durch Kommas getrennte Werte) exportiert. Berichte werden täglich aktualisiert und basieren auf den von Ihnen konfigurierten Objektaltersgruppenfiltern. Bei Verwendung der Amazon-S3-Konsole können Sie die Exportberichtsoption wählen, wenn Sie einen Filter erstellen. Wenn Sie Dateien zum Export auswählen, geben Sie einen Ziel-Bucket und ein optionales Ziel-Präfix an, wohin die Datei geschrieben werden soll. Sie können die Daten in einen Ziel-Bucket in einem anderen Konto exportieren. Der Ziel-Bucket muss sich in derselben Region befinden wie der Bucket, dessen Analyse Sie konfiguriert haben.

Sie müssen eine Bucket-Richtlinie für den Ziel-Bucket erstellen, um Amazon S3 die Berechtigung zu erteilen, zu überprüfen, welcher Eigentümer der Bucket AWS-Konto ist, und Objekte in den Bucket am definierten Speicherort zu schreiben. Eine Beispielrichtlinie finden Sie unter Gewähren von Berechtigungen für S3 Inventory und S3 Analytics.

Nachdem Sie Speicherklassen-Analyseberichte konfiguriert haben, erhalten Sie den exportierten Bericht nach 24 Stunden täglich. Anschließend setzt Amazon S3 die Überwachung fort und stellt tägliche Berichte bereit.

Sie können die CSV-Datei in einer Tabellenkalkulationsanwendung öffnen oder in andere Anwendungen wie Amazon QuickSightimportieren. Informationen zur Verwendung von Amazon S3-

Dateien mit Amazon QuickSightfinden Sie unter Erstellen eines Datensatzes mit Amazon S3-Dateien im Amazon- QuickSight Benutzerhandbuch.

Daten in den exportierten Dateien werden nach dem Datum innerhalb der Objektaltersgruppe sortiert, wie in den folgenden Beispielen gezeigt. Wenn die Speicherklasse STANDARD ist, enthält die Zeile Daten für die Spalten `ObjectAgeForSIATransition` und `RecommendedObjectAgeForSIATransition`.

Date	ConfigId	Filter	StorageClass	ObjectAge	ObjectCount	DataUploaded_MB	Storage_MB	DataRetrieved_MB	GetRequestCount	CumulativeAccessRatio	ObjectAgeForSIATransition	RecommendedObjectAgeForSIATransition
8/17/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
9/2/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/22/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
8/25/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
9/6/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/30/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/28/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/21/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
9/5/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		

Am Ende des Berichts wird die Objektaltersgruppe als ALL angegeben. Die ALL-Zeilen enthalten kumulative Summen, einschließlich Objekte, die kleiner als 128 KB sind, für alle Altersgruppen für diesen Tag.

8/24/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014
9/3/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.02426125	015-029
8/28/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.03545875	015-029
8/17/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014
8/25/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014
9/6/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.0209529	015-029
9/4/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.02304819	015-029
8/22/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014
8/21/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014
8/30/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.03073092	015-029
8/20/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014

Der nächste Abschnitt beschreibt die im Bericht verwendeten Spalten.

Layout der exportierten Datei

In der folgenden Tabelle wird das Layout der exportierten beschrieben.

Konfigurieren der Speicherklassen-Analyse

Unter Verwendung des Tools für die Speicherklassen-Analyse von Amazon S3 können Sie Speicherzugriffsmuster analysieren, anhand derer Sie entscheiden können, ob Sie die richtigen Daten in die richtige Speicherklasse einordnen. Die Speicherklassen-Analyse beobachtet Datenzugriffsmuster analysieren, anhand derer Sie entscheiden können, wann Sie STANDARD-Speicher mit weniger häufigem Zugriff in die Speicherklasse STANDARD_IA (IA für "infrequent access" (seltener Zugriff)) überführen sollen. Weitere Informationen zu STANDARD_IA finden Sie in den Amazon-S3-FAQ und Verwenden von Amazon-S3-Speicherklassen.

Sie können die Speicherklassen-Analyse Einrichten, indem Sie konfigurieren, welche Objektdaten analysiert werden sollen. Sie können die Speicherklassenanalsyse für die folgenden Aufgaben konfigurieren:

- Analyse des gesamten Inhalts eines Buckets.

  Sie erhalten eine Analyse für alle Objekte im Bucket.

- Analyse von nach Präfix und Tags gruppierten Objekten.

  Sie können Filter konfigurieren, um Objekte nach einem gemeinsamen Präfix oder nach Objekt-Markierungen zu gruppieren, oder nach einer Kombination aus Präfix und Markierungen. Sie erhalten für jeden konfigurierten Filter eine separate Analyse. Sie können maximal 1000 Filterkonfigurationen pro Bucket verwenden.

- Export von Analysedaten.

  Wenn Sie die Speicherklassen-Analyse für einen Bucket oder Filter konfigurieren, können Sie die Analysedaten täglich in eine Datei exportieren lassen. Die Analyse für den Tag wird der Datei hinzugefügt, um ein Verlaufsanalyseprotokoll für den konfigurierten Filter zu erstellen. Die Datei wird täglich an dem Ziel Ihrer Wahl aktualisiert. Wenn Sie Dateien zum Export auswählen, geben Sie einen Ziel-Bucket und ein optionales Ziel-Präfix an, wohin die Datei geschrieben werden soll.

Sie können die Amazon S3-Konsole, die REST-API oder die - AWS CLI oder - AWS SDKs verwenden, um die Speicherklassenanalyse zu konfigurieren.

> **Important**
>
> Die Speicherklassen-Analyse gibt keine Empfehlungen für Übergänge in die Speicherklassen ONEZONE_IA oder S3 Glacier Flexible Retrieval.
> Wenn Sie die Speicherklassenanalyse so konfigurieren möchten, dass Ihre Ergebnisse als CSV-Datei exportiert werden, und der Ziel-Bucket die Standard-Bucket-Verschlüsselung mit einem verwendet AWS KMS key, müssen Sie die AWS KMS Schlüsselrichtlinie aktualisieren, um Amazon S3 die Berechtigung zum Verschlüsseln der CSV-Datei zu erteilen. Detaillierte Anweisungen finden Sie unter Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres vom Kunden verwalteten Schlüssels für die Verschlüsselung.

Weitere Informationen zu Analytics finden Sie unter Amazon S3 analytics – Speicherklassen-Analyse.

Verwenden der S3-Konsole

Konfigurieren der Speicherklassen-Analyse

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie die Speicherklassen-Analyse konfigurieren wollen.
3. Wählen Sie den Tab Metrics.
4. Wählen Sie unter Storage Class Analysis (Speicherklassen-Analyse) die Option Create analytics configuration (Analysenkonfiguration erstellen).
5. Geben Sie einen Namen für den Filter ein. Wenn der komplette Bucket analysiert werden soll, tragen Sie nichts in das Feld Prefix (Präfix) ein.
6. Geben Sie im Feld Prefix (Präfix) Text für das Präfix für die Objekte ein, die Sie analysieren möchten.
7. Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Geben Sie einen Schlüssel und Wert für den Tag ein. Sie können ein Präfix und mehrere Markierungen eingeben.
8. Optional können Sie Enable (Aktivieren) unter Export CSV (CSV exportieren) auswählen, um Analyseberichte in einer flachen .csv-Datei (durch Kommas getrennte Werte) zu exportieren. Wählen Sie einen Ziel-Bucket aus, in dem die Datei gespeichert werden kann. Sie können ein Präfix für den Ziel-Bucket eingeben. Der Ziel-Bucket muss sich in derselben befinden AWS-Region wie der Bucket, für den Sie die Analyse einrichten. Der Ziel-Bucket kann sich in einem anderem AWS-Konto befinden.

   Wenn der Ziel-Bucket für die CSV-Datei die Standard-Bucket-Verschlüsselung mit einem KMS-Schlüssel verwendet, müssen Sie die AWS KMS Schlüsselrichtlinie aktualisieren, um Amazon S3 die Berechtigung zum Verschlüsseln der CSV-Datei zu erteilen. Detaillierte Anweisungen finden Sie unter Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres vom Kunden verwalteten Schlüssels für die Verschlüsselung.
9. Wählen Sie Create configuration (Konfiguration erstellen).

Amazon S3 erstellt eine Bucket-Richtlinie für den Ziel-Bucket, mit der Amazon S3 Schreibberechtigung gewährt wird. Auf diese Weise kann es die Exportdaten in den Bucket schreiben.

Wenn beim Erstellen der Bucket-Richtlinie ein Fehjler auftritt, erhalten Sie Anweisungen zur Fehlerbehebung. Wenn Sie beispielsweise einen Ziel-Bucket in einem anderen AWS-Konto ausgewählt haben und nicht über die Berechtigungen verfügen, die Bucket-Richtlinie zu lesen und zu verändern, wird die folgende Meldung angezeigt. Sie müssen den Eigentümer des Ziel-Buckets auffordern, dem Ziel-Bucket die angezeigte Bucket-Richtlinie hinzuzufügen. Wird die Richtlinie dem Ziel-Bucket nicht hinzugefügt, erhalten Sie keine Exportdaten, weil Amazon S3 nicht die Berechtigung besitzt, in den Ziel-Bucket zu schreiben. Wenn der Quell-Bucket nicht dem Konto des aktuellen Benutzers gehört, muss die richtige Konto-ID des Quell-Buckets in der Richtlinie ersetzt werden.

Informationen über die exportierten Daten und die Funktionsweise des Filters finden Sie unter Amazon S3 analytics – Speicherklassen-Analyse.

Verwenden der REST-API

Um die Speicherklassenanalyse mithilfe der REST-API zu konfigurieren, verwenden Sie die PutBucketAnalyticsConfiguration. Sie können den entsprechenden Vorgang auch mit der AWS CLI oder AWS SDKs verwenden.

Sie können die folgenden REST-APIs verwenden, um mit Storage Class Analysis zu arbeiten:

- DELETE Bucket Analytics configuration
- GET Bucket Analytics configuration
- List Bucket Analytics Configuration

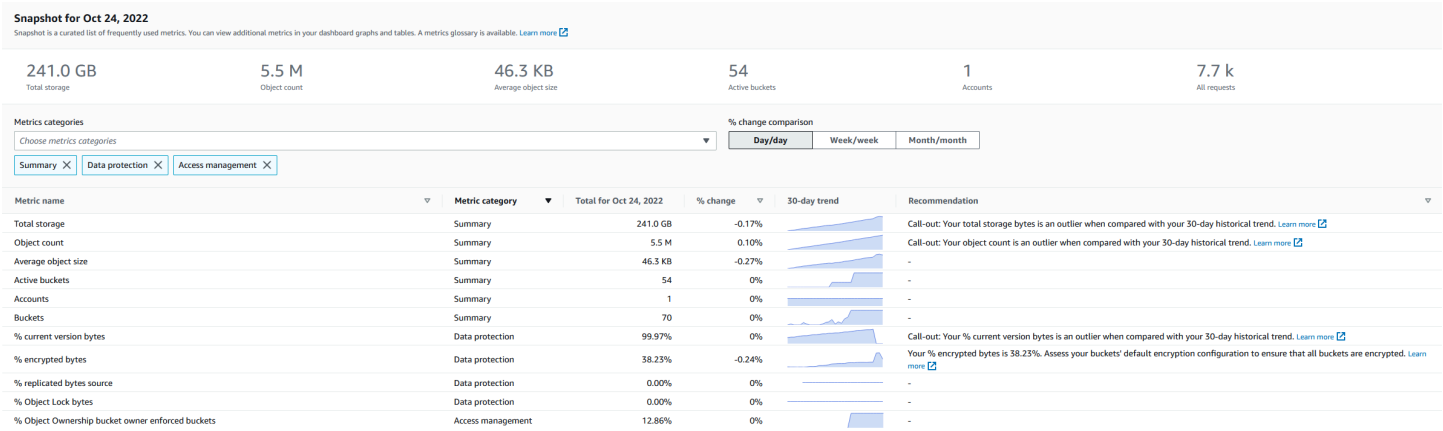
Bewerten Ihrer Speicheraktivität und -nutzung mit Amazon S3 Storage Lens

Amazon S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Erkenntnisse zu Objektspeichern und Aktivitäten erhalten können. S3 Storage Lens analysiert Metriken, um kontextbezogene Empfehlungen zur Optimierung der Speicherkosten und zur Anwendung bewährter Datenschutzmethoden zu geben.

Mithilfe der Metriken von S3 Storage Lens lässt sich eine Zusammenfassung der Erkenntnisse generieren. Beispielseise können Sie herausfinden, wie viel Speicher Sie in der gesamten Organisation haben oder welche Buckets und Präfixe am schnellsten wachsen. Außerdem können Sie anhand der Metriken von S3 Storage Lens umfassende Möglichkeiten zur Kostenoptimierung aufdecken, bewährte Methoden für den Datenschutz und die Zugriffsverwaltung implementieren und die Leistung von Anwendungs-Workloads verbessern. Sie können beispielsweise Buckets

identifizieren, für die keine S3-Lebenszyklusregeln gelten, damit unvollständige mehrteilige Uploads, die älter als 7 Tage sind, abgebrochen werden. Sie können auch Buckets identifizieren, die nicht den bewährten Datenschutzmethoden entsprechen, z. B. die Verwendung von S3 Replication oder S3 Versionierung.

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.



Metriken und Funktionen von S3 Storage Lens

S3 Storage Lens bietet ein interaktives Standard-Dashboard, das täglich aktualisiert wird. Gemäß Vorkonfiguration von S3 Storage Lens werden in diesem Dashboard die zusammengefassten Erkenntnisse und Trends Ihres gesamten Kontos visualisiert und täglich in der S3-Konsole aktualisiert. Metriken aus diesem Dashboard werden auch in Ihrem Konto-Snapshot auf der Seite Buckets zusammengefasst. Weitere Informationen finden Sie unter Standard-Dashboard.

Wenn Sie weitere Dashboards erstellen und ihren Umfang nach AWS-Regionen, S3-Buckets oder Konten (für AWS Organizations) festlegen möchten, erstellen Sie eine Dashboard-Konfiguration für S3 Storage Lens. Sie können Dashboard-Konfigurationen von S3 Storage Lens mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI), der AWS SDKs oder der Amazon-S3-REST-API erstellen und verwalten. Wenn Sie ein S3-Storage-Lens-Dashboard erstellen oder bearbeiten, definieren Sie den Dashboard-Umfang und die Metrikauswahl.

S3 Storage Lens bietet neben kostenlosen Metriken auch erweiterte Metriken und Empfehlungen, auf die Sie gegen eine zusätzliche Gebühr upgraden können. Mit erweiterten Metriken und Empfehlungen können Sie auf zusätzliche Metriken und Funktionen zugreifen, um Erkenntnisse über Ihren Speicher zu erhalten. Zu diesen Funktionen gehören erweiterte Metrikkategorien, Präfixaggregation, kontextbezogene Empfehlungen und Amazon-CloudWatch-Veröffentlichung. Präfixaggregation und kontextbezogene Empfehlungen sind nur in der Amazon-S3-Konsole verfügbar. Weitere Informationen zu den Preisen für S3 Storage Lens finden Sie unter Amazon S3-Preise.

Metrikkategorien

In den kostenlosen und erweiterten Stufen sind die Metriken in Kategorien unterteilt, die auf wichtige Anwendungsfälle wie Kostenoptimierung und Datenschutz abgestimmt sind. Zu den kostenlosen Metriken gehören Zusammenfassung, Kostenoptimierung, Datenschutz, Zugriffsverwaltung, Leistung und Ereignismetriken. Wenn Sie auf erweiterte Metriken und Empfehlungen upgraden, können Sie erweiterte Metriken zur Kostenoptimierung und zum Datenschutz aktivieren. Anhand dieser erweiterten Metriken können Sie Ihre S3-Speicherkosten weiter senken und Ihre Datenschutzpolitik verbessern. Sie können auch Aktivitätsmetriken und detaillierte Statuscode-Metriken aktivieren, um die Leistung von Anwendungs-Workloads zu verbessern, die auf Ihre S3-Buckets zugreifen. Weitere Informationen zu kostenlosen und erweiterten Metriken finden Sie unter Metrikauswahl.

Sie können Ihren Speicher auf der Grundlage bewährter Methoden von S3 bewerten, z. B. indem Sie den Prozentsatz der Buckets analysieren, für die die Verschlüsselung, S3 Object Lock oder S3 Versioning aktiviert ist. Sie können auch potenzielle Kosteneinsparungsmöglichkeiten identifizieren. Zum Beispiel können Sie Metriken zur Anzahl von S3-Lebenszyklusregeln verwenden, um Buckets zu identifizieren, in denen Lebenszyklusregeln zum Ablauf und Übergang fehlen. Sie können auch Ihre Anforderungsaktivität pro Bucket analysieren, um Buckets zu finden, in denen Objekte in eine kostengünstigere Speicherklasse überführt werden können. Weitere Informationen finden Sie unter Anwendungsfälle für Metriken von Amazon S3 Storage Lens.

Metrik-Export

Zusätzlich zur Ansicht des Dashboards in der S3-Konsole können Sie Metriken im CSV- oder Parquet-Format zur weiteren Analyse mit dem Analysetool Ihrer Wahl in einen S3-Bucket exportieren. Weitere Informationen finden Sie unter Anzeigen von Amazon S3-Storage-Lens-Metriken mit einem Datenexport.

Amazon-CloudWatch-Veröffentlichung

Sie können S3-Storage-Lens-Nutzungs- und Aktivitätsmetriken auf Amazon CloudWatch veröffentlichen, um eine einheitliche Ansicht Ihres Betriebszustands in CloudWatch-Dashboards zu erstellen. Sie können auch CloudWatch-Funktionen wie Alarme und ausgelöste Aktionen, Metrikmathematik und Anomalieerkennung verwenden, um S3-Storage-Lens-Metriken zu überwachen und Maßnahmen zu ergreifen. Darüber hinaus ermöglichen CloudWatch-API-Operationen Anwendungen, einschließlich Drittanbietern, den Zugriff auf Ihre S3-Storage-Lens-Metriken. Die CloudWatch-Veröffentlichungsoption ist für Dashboards verfügbar, die auf erweiterte Metriken und Empfehlungen von S3 Storage Lens aktualisiert wurden. Weitere Informationen zur Unterstützung von S3-Storage-Lens-Metriken in CloudWatch finden Sie unter Überwachen von Metriken von S3 Storage Lens in CloudWatch.

Weitere Informationen zur Verwendung von S3 Storage Lens finden Sie in den folgenden Themen.

Themen

Grundlegendes zu Amazon S3 Storage Lens

Important

Amazon S3 wendet jetzt serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüssel (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Ab dem 5. Januar 2023 werden alle neuen Objekt-Uploads auf Amazon S3 ohne zusätzliche Kosten und ohne Auswirkungen auf die Leistung automatisch verschlüsselt. Der Status der automatischen Verschlüsselung für die Standardverschlüsselungskonfiguration von S3-Buckets und für neue Objekt-Uploads ist in AWS CloudTrail-Protokollen, S3 Inventory, S3 Storage Lens, der Amazon-S3-Konsole und als zusätzlicher Amazon-S3-API-

Antwortheader in der AWS Command Line Interface und den AWS-SDKs verfügbar. Weitere Informationen finden Sie unter Häufig gestellte Fragen zur Standardverschlüsselung.

Amazon S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Sie können Metriken von S3 Storage Lens verwenden, um zusammenfassende Erkenntnisse zu gewinnen und z. B. herauszufinden, wie viel Speicher Sie in Ihrer gesamten Organisation haben oder welche Buckets und Präfixe am schnellsten wachsen. Außerdem können Sie anhand der Metriken von S3 Storage Lens umfassende Möglichkeiten zur Kostenoptimierung aufdecken, bewährte Methoden für den Datenschutz implementieren und die Leistung von Anwendungs-Workloads verbessern. Sie können beispielsweise Buckets identifizieren, für die keine S3-Lebenszyklusregeln gelten, damit unvollständige mehrteilige Uploads, die älter als 7 Tage sind, ablaufen. Sie können auch Buckets identifizieren, die nicht den bewährten Datenschutzmethoden entsprechen, z. B. die Verwendung von S3 Replication oder S3 Versionierung. S3 Storage Lens analysiert Metriken, um kontextbezogene Empfehlungen zur Optimierung der Speicherkosten und zur Anwendung bewährter Datenschutzmethoden zu geben.

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden. Sie können S3-Storage-Lens-Dashobards mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI), der AWS SDKs oder der Amazon-S3-REST-API erstellen und verwalten.

Konzepte und Terminologie von S3 Storage Lens

Dieser Abschnitt enthält die Terminologie und Konzepte, die für das Verständnis und die erfolgreiche Verwendung von Amazon S3 Storage Lens unerlässlich sind.

Themen

- Dashboards
- Konto-Snapshot
- Metrik-Export
- Heimatregion
- Aufbewahrungszeitraum
- Metrikkategorien
- Empfehlungen
- Metrikauswahl
- S3 Storage Lens und AWS Organizations

Konfiguration des Dashboards

S3 Storage Lens erfordert eine Dashboard-Konfiguration, die die Eigenschaften enthält, mit denen Metriken in Ihrem Namen für ein einzelnes Dashboard oder einen Export aggregiert werden. Wenn Sie eine Konfiguration erstellen, wählen Sie den Dashboard-Namen und die Heimatregion aus, die Sie nach der Erstellung des Dashboards nicht mehr ändern können. Sie können optional Tags hinzufügen und einen Metrikexport im CSV- oder Parquet-Format konfigurieren.

In der Dashboard-Konfiguration definieren Sie auch den Dashboard-Umfang und die Metrikauswahl. Der Umfang kann den gesamten Speicherplatz für Ihr Organisationskonto oder Abschnitte umfassen, die nach Region, Bucket und Konto gefiltert sind. Wenn Sie die Metrikauswahl konfigurieren, wählen Sie zwischen kostenlosen Metriken und erweiterten Metriken und Empfehlungen, auf die Sie gegen eine zusätzliche Gebühr upgraden können. Mit erweiterten Metriken und Empfehlungen können Sie auf zusätzliche Metriken und Funktionen zugreifen. Zu diesen Funktionen gehören erweiterte Metrikkategorien, Aggregation auf Präfixebene, kontextbezogene Empfehlungen und Amazon-CloudWatch-Veröffentlichung. Weitere Informationen zu den Preisen für S3 Storage Lens finden Sie unter Amazon S3-Preise.

Standard-Dashboard

Das Standard-Dashboard von S3 Storage Lens in der Konsole heißt default-account-dashboard. Gemäß S3-Vorkonfiguration werden in diesem Dashboard die zusammengefassten Erkenntnisse und Trends Ihres gesamten Kontos visualisiert und täglich in der S3-Konsole aktualisiert. Sie können den Konfigurationsbereich des Standard-Dashboards zwar nicht ändern, haben aber die Möglichkeit, ein Upgrade für die Metrikauswahl durchzuführen und statt der kostenlosen Metriken die erweiterten

Metriken und Empfehlungen zu verwenden. Sie können den optionalen Metrikexport konfigurieren oder sogar das Dashboard deaktivieren. Löschen können Sie das Standard-Dashboard jedoch nicht.

> **Note**
>
> Wenn Sie Ihr Standard-Dashboard deaktivieren, wird es nicht mehr aktualisiert. Sie erhalten auf Ihrem S3 Storage Lens-Dashboard, in Ihrem Metrikexport oder im Konto-Snapshot auf der S3-Seite Buckets keine neuen täglichen Metriken mehr. Wenn Ihr Dashboard erweiterte Metriken und Empfehlungen verwendet, werden Ihnen keine Gebühren mehr berechnet. Sie können im Dashboard weiterhin historische Daten anzeigen, bis der Zeitraum von 14 Tagen für Datenabfragen abläuft. Wenn Sie erweiterte Metriken und Empfehlungen aktiviert haben, beträgt dieser Zeitraum 15 Monate. Sie können das Dashboard innerhalb des Ablaufzeitraums wieder aktivieren, um auf historische Daten zuzugreifen.

Dashboards

Sie können zusätzliche S3-Storage-Lens-Dashboards erstellen und ihren Umfang nach AWS-Regionen, S3-Buckets oder Konten (für AWS Organizations) festlegen. Wenn Sie ein S3-Storage-Lens-Dashboard erstellen oder bearbeiten, definieren Sie den Dashboard-Umfang und die Metrikauswahl. S3 Storage Lens bietet neben kostenlosen Metriken auch erweiterte Metriken und Empfehlungen, auf die Sie gegen eine zusätzliche Gebühr upgraden können. Mit erweiterten Metriken und Empfehlungen können Sie auf zusätzliche Metriken und Funktionen zugreifen, um Erkenntnisse über Ihren Speicher zu erhalten. Dazu gehören erweiterte Metrikkategorien, Aggregation auf Präfixebene, kontextbezogene Empfehlungen und Amazon-CloudWatch-Veröffentlichung. Weitere Informationen zu den Preisen für S3 Storage Lens finden Sie unter Amazon S3-Preise.

Sie können das Dashboard auch deaktivieren oder löschen. Wenn Sie ein Dashboard deaktivieren, wird es nicht mehr aktualisiert und Sie erhalten keine neuen täglichen Metriken mehr. Sie können historische Daten dann noch bis zum Ende des Ablaufzeitraums von 14 Tagen einsehen. Wenn Sie erweiterte Metriken und Empfehlungen für das Dashboard aktiviert haben, beträgt dieser Zeitraum 15 Monate. Sie können das Dashboard innerhalb des Ablaufzeitraums wieder aktivieren, um auf historische Daten zuzugreifen.

Wenn Sie ein Dashboard löschen, gehen alle Dashboard-Konfigurationseinstellungen verloren. Sie erhalten dann keine neuen täglichen Metriken mehr und können nicht mehr auf die mit dem Dashboard verbundenen historischen Daten zugreifen. Wenn Sie auf die historischen Daten eines

gelöschten Dashboards zugreifen möchten, müssen Sie ein neues Dashboard mit demselben Namen in derselben Heimatregion erstellen.

> **Note**
>
> - Sie können S3 Storage Lens verwenden, um bis zu 50 Dashboards pro Heimatregion zu erstellen.
> - Dashboards auf Organisationsebene können nur auf einen regionalen Bereich beschränkt werden.

Konto-Snapshot

Der Konto-Snapshot von S3 Storage Lens fasst Metriken aus Ihrem Standard-Dashboard zusammen und zeigt Ihren Gesamtspeicher, die Objektzahl und die durchschnittliche Objektgröße auf der Seite Buckets der S3-Konsole an. Mit diesem Konto-Snapshot haben Sie schnellen Zugriff auf Erkenntnisse über Ihren Speicher, ohne die Seite Buckets verlassen zu müssen. Der Konto-Snapshot bietet auch mit einem Klick Zugriff auf Ihr interaktives S3-Storage-Lens-Dashboard.

Sie können Ihr Dashboard verwenden, um Erkenntnisse und Trends zu visualisieren, Ausreißer zu kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten und zur Anwendung bewährter Datenschutzmethoden einzusehen. Ihr Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, Bucket-, Objekt- oder sogar Präfixebene zu generieren. Sie können auch einmal täglich einen Metrikexport in einen S3-Bucket im CSV- oder Parquet-Format senden.

Sie können den Dashboard-Umfang des default-account dashboard (Standardkonto-Dashboards) nicht ändern, da er mit dem Account snapshot (Konto-Snapshot) verknüpft ist. Sie können jedoch die Auswahl der Metriken in Ihrem default-account-dashboard (Standardkonto-Dashboard) von kostenlosen Metriken auf kostenpflichtige erweiterte Metriken und Empfehlungen upgraden. Nach dem Upgrade können Sie alle Anfragen, hochgeladenen Bytes und heruntergeladenen Bytes im Account snapshot (Konto-Snapshot) von S3 Storage Lens anzeigen.

> **Note**
>
> Wenn Sie Ihr Standard-Dashboard deaktivieren, wird Ihr Account snapshot (Konto-Snapshot) nicht mehr aktualisiert. Sie können das default-account-dashboard (Standardkonto-

Dashboard) erneut aktivieren, um Metriken weiterhin im Account snapshot (Konto-Snapshot) anzuzeigen.

Metrik-Export

Ein Metrik-Export ist eine Datei, die alle in Ihrer S3-Storage-Lens-Konfiguration erfassten Metriken enthält. Diese Informationen werden täglich im CSV- oder Parquet-Format generiert und an einen S3-Bucket gesendet. Sie können den Metrikexport für weitere Analysen verwenden, indem Sie das Metrik-Tool Ihrer Wahl verwenden. Der S3-Bucket für den Export Ihrer Metriken muss sich in derselben Region wie Ihre S3-Storage-Lens-Konfiguration befinden. Sie können einen Metrikexport von S3 Storage Lens über die S3-Konsole generieren, indem Sie Ihre Dashboard-Konfiguration bearbeiten. Einen Metrikexport können Sie auch mithilfe der AWS CLI und AWS SDKs konfigurieren.

Heimatregion

Die Heimatregion ist die AWS-Region, in der alle S3-Storage-Lens-Metriken für eine bestimmte Dashboard-Konfiguration gespeichert werden. Sie müssen eine Heimatregion auswählen, wenn Sie Ihr S3-Storage-Lens-Dashboard konfigurieren. Nachdem Sie eine Heimatregion ausgewählt haben, können Sie sie nicht mehr ändern. Wenn Sie eine Storage-Lens-Gruppe erstellen, empfehlen wir außerdem, dieselbe Heimatregion wie für das Storage-Lens-Dashboard auszuwählen.

Note

Sie können eine der folgenden Regionen als Heimatregion auswählen:

- US East (N. Virginia) – `us-east-1`
- US East (Ohio) – `us-east-2`
- US West (N. California) – `us-west-1`
- US West (Oregon) – `us-west-2`
- Asia Pacific (Mumbai) – `ap-south-1`
- Asien-Pazifik (Seoul) – `ap-northeast-2`
- Asia Pacific (Singapore) – `ap-southeast-1`
- Asien-Pazifik (Sydney) – `ap-southeast-2`
- Asien-Pazifik (Tokio) – `ap-northeast-1`
- Canada (Central) – `ca-central-1`
- China (Peking) – `cn-north-1`

- China (Ningxia) – `cn-northwest-1`
- Europe (Frankfurt) – `eu-central-1`
- Europe (Ireland) – `eu-west-1`
- Europe (London) – `eu-west-2`
- Europe (Paris) – `eu-west-3`
- Europe (Stockholm) – `eu-north-1`
- South America (São Paulo) – `sa-east-1`

Aufbewahrungszeitraum

S3-Storage-Lens-Metriken werden beibehalten, damit Sie historische Trends erkennen und Unterschiede bei Speichernutzung und -aktivität im Zeitverlauf vergleichen können. Sie können Metriken von Amazon S3 Storage Lens für Abfragen verwenden, um historische Trends anzuzeigen und Unterschiede in Ihrer Speichernutzung und -aktivität im Zeitverlauf zu vergleichen.

Alle Metriken für S3 Storage Lens werden für einen Zeitraum von 15 Monaten aufbewahrt. Metriken sind jedoch nur für Abfragen für eine bestimmte Dauer verfügbar, die von Ihrer Metrikauswahl abhängt. Diese Dauer kann nicht geändert werden. Kostenlose Metriken stehen für Abfragen für einen Zeitraum von 14 Tagen zur Verfügung, und fortschrittliche Metriken stehen für Abfragen für 15 Monate zur Verfügung.

Metrikkategorien

In den kostenlosen und erweiterten Stufen sind die Metriken von S3 Storage Lens in Kategorien unterteilt, die auf wichtige Anwendungsfälle wie Kostenoptimierung und Datenschutz abgestimmt sind. Zu den kostenlosen Metriken gehören Zusammenfassung, Kostenoptimierung, Datenschutz, Zugriffsverwaltung, Leistung und Ereignismetriken. Wenn Sie auf erweiterte Metriken und Empfehlungen upgraden, können Sie zusätzliche Metriken zur Kostenoptimierung und zum Datenschutz aktivieren, mit denen Sie Ihre S3-Speicherkosten weiter senken und den Datenschutz gewährleisten können. Sie können auch Aktivitätsmetriken und detaillierte Statuscode-Metriken aktivieren, um die Leistung von Anwendungs-Workloads zu verbessern.

Die folgende Liste enthält alle kostenlosen und erweiterten Metrikkategorien. Eine vollständige Liste der einzelnen Metriken, die in jeder Kategorie enthalten sind, finden Sie im Metrik-Glossar.

Zusammenfassende Metriken

Zusammenfassende Metriken bieten allgemeine Erkenntnisse über Ihren S3-Speicher, einschließlich Ihrer gesamten Speicherbytes und der Anzahl der Objekte.

Metriken zur Kostenoptimierung

Metriken zur Kostenoptimierung liefern Erkenntnisse, die Sie zur Verwaltung und Optimierung Ihrer Speicherkosten nutzen können. Sie können beispielsweise Buckets identifizieren, unvollständige mehrteilige Uploads enthalten, die älter als 7 Tage sind.

Mit erweiterten Metriken und Empfehlungen können Sie erweiterte Metriken zur Kostenoptimierung aktivieren. Zu diesen Metriken gehören Metriken für die Anzahl der S3-Lebenszyklusregeln, mit denen Sie die Anzahl der S3-Lebenszyklusregeln pro Bucket für den Ablauf und für den Übergang ermitteln können.

Metriken zum Datenschutz

Metriken zum Datenschutz liefern Erkenntnisse über Datenschutzfunktionen wie Verschlüsselung und S3 Versioning. Sie können diese Metriken verwenden, um Buckets zu identifizieren, die nicht den bewährten Datenschutzmethoden entsprechen. Sie können beispielsweise Buckets identifizieren, die keine Standardverschlüsselung mit AWS Key Management Service-Schlüsseln (SSE-KMS) oder S3 Versioning verwenden.

Mit erweiterten Metriken und Empfehlungen können Sie erweiterte Metriken zum Datenschutz aktivieren. Zu diesen Metriken gehören Metriken für die Anzahl der Replikationsregeln pro Bucket.

Metriken zur Zugriffsverwaltung

Metriken zur Zugriffsverwaltung liefern Erkenntnisse im Hinblick auf S3 Object Ownership. Sie können diese Metriken verwenden, um zu sehen, welche Einstellungen Ihre Buckets für die Objekteigentümerschaft verwenden.

Ereignismetriken

Ereignismetriken bieten Einblicke in S3-Ereignisbenachrichtigungen. Mithilfe von Ereignismetriken können Sie sehen, für welche Buckets S3-Ereignisbenachrichtigungen konfiguriert sind.

Leistungsmetriken

Leistungsmetriken liefern relevante Erkenntnisse für S3 Transfer Acceleration. Anhand von Leistungsmetriken können Sie sehen, für welche Buckets Transfer Acceleration aktiviert ist.

Aktivitätsmetriken (erweitert)

Wenn Sie das Dashboard auf erweiterte Metriken und Empfehlungen upgraden, können Sie Aktivitätsmetriken aktivieren. Aktivitätsmetriken liefern Details darüber, wie Ihr Speicherplatz angefordert wird (z. B. All-Anforderungen, Get-Anforderungen, Put-Anforderungen), sowie über hoch- oder heruntergeladene Bytes und Fehler.

Anhand von Aktivitätsmetriken auf Präfixebene können Sie ermitteln, welche Präfixe selten verwendet werden, sodass Sie im S3-Lebenszyklus zu einer optimaleren Speicherklasse wechseln können.

Detaillierte Statuscode-Metriken (erweitert)

Wenn Sie das Dashboard auf erweiterte Metriken und Empfehlungen upgraden, können Sie detaillierte Statuscode-Metriken aktivieren. Detaillierte Statuscode-Metriken bieten Einblicke in HTTP-Statuscodes wie 403 Forbidden und 503 Service Unavailable, die Sie zur Behebung von Zugriffs- oder Leistungsproblemen nutzen können. Sie können sich beispielsweise die Metrik 403 Forbidden error count (Anzahl der Fehler 403 Forbidden) ansehen, um Workloads zu identifizieren, die auf Buckets zugreifen, ohne dass die korrekten Berechtigungen angewendet wurden.

Detaillierte Statuscode-Metriken auf Präfixebene können verwendet werden, um ein besseres Verständnis der Vorkommen des HTTP-Statuscodes nach Präfix zu erhalten. Mit den Metriken zur Anzahl der 503-Fehler können Sie beispielsweise Präfixe ermitteln, die während der Datenerfassung Drosselungsanfragen erhalten.

Empfehlungen

S3 Storage Lens bietet automatisierte Empfehlungen, die Sie bei der Optimierung Ihres Speichers unterstützen. Die Empfehlungen werden kontextbasiert neben relevanten Metriken im S3-Storage-Lens-Dashboard platziert. Historische Daten kommen nicht für Empfehlungen in Betracht, da Empfehlungen nur für aktuelle Geschehnisse relevant sind. Empfehlungen werden nur angezeigt, wenn sie relevant sind.

Es gibt folgende Arten von S3-Storage-Lens-Empfehlungen:

- Vorschläge

  Vorschläge weisen Sie auf Trends beim Speicher und bei Aktivitäten hin, die auf Möglichkeiten zur Optimierung der Speicherkosten oder auf bewährte Datenschutzmethoden hindeuten könnten. Anhand der vorgeschlagenen Themen können Sie im Amazon-S3-Benutzerhandbuch und im S3-Storage-Lens-Dashboard nach weiteren Informationen zu den spezifischen Regionen, Buckets oder Präfixen suchen.

- Callouts

  Callouts sind Empfehlungen, die Sie auf interessante Anomalien in Bezug auf den Speicher und Aktivitäten aufmerksam machen, die über einen bestimmten Zeitraum auftreten und möglicherweise Maßnahmen erfordern oder überwacht werden sollten.

  - Ausreißer-Callouts

    S3 Storage Lens bietet Callouts für Metriken, bei denen es sich auf der Basis Ihres Trends der letzten 30 Tage um Ausreißer handelt. Der Ausreißer wird mit einem Standardwert berechnet, der auch als Z-Score bezeichnet wird. Bei diesem Score wird die Metrik des aktuellen Tages vom Durchschnitt der letzten 30 Tage für die Metrik subtrahiert. Die Metrik des aktuellen Tages wird dann durch die Standardabweichung für die Metrik in den letzten 30 Tagen dividiert. Der resultierende Score liegt normalerweise zwischen -3 und +3. Diese Zahl stellt die Anzahl der Standardabweichungen dar, die die Metrik des aktuellen Tages vom Mittelwert entfernt ist.

    Metriken mit einem Score von > 2 oder < -2 werden in S3 Storage Lens als Ausreißer erachtet, da sie über oder unter dem Wert von 95 Prozent der normal verteilten Daten liegen.

  - Callouts für signifikante Änderungen

    Das Callout für signifikante Änderungen gilt für Metriken, die sich erwartungsgemäß selten ändern. Daher ist er auf eine höhere Sensitivität eingestellt als die Ausreißerberechnung, die sich typischerweise im Bereich von +/-20 Prozent gegenüber dem Vortag, der Vorwoche oder dem Vormonat befindet.

    Bearbeiten von Callouts für den Speicher und Aktivitäten – Wenn Sie ein Callout für signifikante Änderungen erhalten, muss das nicht auf ein Problem hinweisen. Das Callout könnte das Ergebnis einer erwarteten Änderung des Speichers sein. Beispielsweise können Sie vor Kurzem eine große Anzahl neuer Objekte hinzugefügt, eine große Anzahl von Objekten gelöscht oder ähnliche geplante Änderungen vorgenommen haben.

    Wenn Sie in Ihrem Dashboard ein Callout für signifikante Änderungen sehen, sollten Sie zunächst bestimmen, ob es durch die jüngsten Umstände erklärt werden kann. Ist dies nicht der Fall, können Sie über das S3-Storage-Lens-Dashboard weitere Details abrufen, um herauszufinden, welchen spezifischen Regionen, Buckets oder Präfixen die Fluktuation zugrunde liegt.

- Erinnerungen

Erinnerungen bieten Einblicke in die Funktionsweise von Amazon S3. Sie können Ihnen helfen, mehr darüber zu erfahren, wie Sie S3-Funktionen nutzen können, um die Speicherkosten zu senken oder bewährte Datenschutzmethoden anzuwenden.

Metrikauswahl

S3 Storage Lens bietet zwei Arten von Metriken, die Sie für das Dashboard und den Export auswählen können: kostenlose Metriken und fortschrittliche Metriken und Empfehlungen.

- Kostenlose Metriken

  S3 Storage Lens bietet kostenlose Metriken für alle Dashboards und Konfigurationen. Kostenlose Metriken enthalten relevante Metriken für Ihre Speichernutzung, z. B. die Anzahl der Buckets und der Objekte in Ihrem Konto. Zu den kostenlosen Metriken gehören auch auf Anwendungsfällen basierende Metriken (z. B. Metriken zur Kostenoptimierung und zum Datenschutz), anhand derer Sie untersuchen können, ob Ihr Speicher gemäß den bewährten Methoden von S3 konfiguriert ist. Alle kostenlosen Metriken werden täglich gesammelt. Daten stehen 14 Tage für Abfragen zur Verfügung. Weitere Informationen darüber, welche Metriken kostenlos sind, finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

- Fortschrittliche Metriken und Empfehlungen

  S3 Storage Lens bietet kostenlose Metriken für alle Dashboards und Konfigurationen mit der Option zum Upgrade auf fortschrittliche Metriken und Empfehlungen. Es fallen zusätzliche Gebühren an. Weitere Informationen finden Sie unter Amazon S3 – Preise.

  Zu den erweiterten Metriken und Empfehlungen gehören neben den kostenlosen Metriken weitere Metriken, wie erweiterte Metriken zum Datenschutz und zur Kostenoptimierung, Aktivitätsmetriken und detaillierte Statuscode-Metriken. Erweiterte Metriken und Empfehlungen umfassen auch Empfehlungen zur Optimierung Ihrer Speichernutzung. Die Empfehlungen werden kontextbasiert neben relevanten Metriken im Dashboard platziert.

  Erweiterte Metriken und Empfehlungen umfassen die folgenden Funktionen:

  - Erweiterte Metriken – Generieren Sie zusätzliche Metriken. Eine vollständige Liste der erweiterten Metrikkategorien finden Sie unter Metrikkategorien. Eine vollständige Liste der Metriken finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.
  - Amazon-CloudWatch-Veröffentlichung – Metriken von S3 Storage Lens werden in CloudWatch veröffentlicht, um eine einheitliche Ansicht des Betriebszustands in CloudWatch-Dashboards zu

erstellen. Sie können auch API-Operationen und Funktionen von CloudWatch wie Alarme und ausgelöste Aktionen, Metrikmathematik und Anomalieerkennung verwenden, um Metriken von S3 Storage Lens zu überwachen und Maßnahmen zu ergreifen. Weitere Informationen finden Sie unter Überwachen von Metriken von S3 Storage Lens in CloudWatch.

- Präfixaggregation – Metriken werden auf Präfixebene gesammelt. Durch die Aktivierung der Präfixaggregation werden alle in der Dashboard-Konfiguration enthaltenen Metriken auf Präfixebene erweitert. Metriken werden nur für Präfixe generiert, die den konfigurierten Schwellenwert erreichen. Beachten Sie, dass Metriken, die auf Präfixebene gelten, mit Präfixaggregation verfügbar sind. Ausgenommen sind Einstellungen auf Bucket-Ebene und Metriken zur Regelanzahl. Metriken auf Präfixebene werden nicht in CloudWatch veröffentlicht.
- Aggregation von Storage-Lens-Gruppen – Metriken werden auf Storage-Lens-Gruppenebene gesammelt. Nachdem Sie Erweiterte Metriken und Empfehlungen und Aggregation von Storage-Lens-Gruppen aktiviert haben, können Sie angeben, welche Storage-Lens-Gruppen in das Storage-Lens-Dashboard aufgenommen oder daraus ausgeschlossen werden sollen. Es muss mindestens eine Storage-Lens-Gruppe angegeben werden. Die angegebenen Storage-Lens-Gruppen müssen sich in der angegebenen Heimatregion des Dashboard-Kontos befinden. Metriken auf Storage-Lens-Gruppenebene werden nicht in CloudWatch veröffentlicht.

Alle fortschrittlichen Metriken werden täglich gesammelt. Daten stehen für Abfragen 15 Monate lang zur Verfügung. Weitere Informationen zu den Speichermetriken, die von S3 Storage Lens aggregiert werden, finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

> **Note**
>
> Empfehlungen sind nur über das S3-Storage-Lens-Dashboard in der Amazon-S3-Konsole verfügbar.

S3 Storage Lens und AWS Organizations

AWS Organizations ist ein AWS-Service, mit dem Sie alle Ihre AWS-Konten in einer Organisationshierarchie zusammenfassen können. In Kombination mit AWS Organizations bietet Amazon S3 Storage Lens eine zentrale Sicht auf Objektspeicher und Aktivitäten in Ihrem gesamten Amazon-S3-Speicher.

Weitere Informationen finden Sie unter Verwenden von Amazon S3 Storage Lens mit AWS Organizations.

- Vertrauenswürdiger Zugriff

  Sie müssen den vertrauenswürdigen Zugriff im Verwaltungskonto Ihrer Organisation aktivieren, damit S3 Storage Lens Speichermetriken und Nutzungsdaten für alle Mitgliedskonten in Ihrer Organisation aggregieren kann. Dann können Sie über das Verwaltungskonto Dashboards oder Exporte für Ihre Organisation erstellen oder delegierten Administratoren Zugriff auf andere Konten in der Organisation gewähren.

  Sie können den vertrauenswürdigen Zugriff für S3 Storage Lens jederzeit deaktivieren, um die Aggregation von Metriken für Ihre Organisation zu beenden.

- Delegated Administrator

  Sie können S3-Storage-Lens-Dashboards und -Metriken für Ihre Organisation mit Ihrem AWS Organizations-Verwaltungskonto erstellen oder delegierten Administrator-Zugriff auf andere Konten in Ihrer Organisation gewähren. Sie können delegierte Administratoren jederzeit abmelden. Durch Abmelden eines delegierten Administrators wird die Aggregation neuer Speichermetriken durch alle Dashboards auf Organisationsebene, die von diesem delegierten Administrator erstellt wurden, automatisch beendet.

Weitere Informationen finden Sie unter Amazon S3 Storage Lens und AWS Organizations im AWS Organizations-Benutzerhandbuch.

Serviceverknüpfte Rollen für Amazon S3 Storage Lens

Neben dem vertrauenswürdigen Zugriff von AWS Organizations Organizations nutzt Amazon S3 Storage Lens AWS Identity and Access Management (IAM) serviceverknüpfte Rollen. Eine serviceverknüpfte Rolle ist ein spezieller Typ von IAM-Rolle, die direkt mit S3 Storage Lens verknüpft ist. Serviceverknüpfte Rollen werden von S3 Storage Lens vordefiniert und enthalten alle Berechtigungen, die zum Erfassen der täglichen Speicher- und Aktivitätsmetriken aus Mitgliedskonten in Ihrer Organisation benötigt werden.

Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für Amazon S3 Storage Lens.

Verwenden von Amazon S3 Storage Lens mit AWS Organizations

Amazon S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Sie können Metriken

von S3 Storage Lens verwenden, um zusammenfassende Erkenntnisse zu gewinnen und z. B. herauszufinden, wie viel Speicher Sie in Ihrer gesamten Organisation haben oder welche Buckets und Präfixe am schnellsten wachsen. Außerdem können Sie anhand der Metriken von S3 Storage Lens umfassende Möglichkeiten zur Kostenoptimierung aufdecken, bewährte Methoden für den Datenschutz implementieren und die Leistung von Anwendungs-Workloads verbessern. Sie können beispielsweise Buckets identifizieren, für die keine S3-Lebenszyklusregeln gelten, damit unvollständige mehrteilige Uploads, die älter als 7 Tage sind, ablaufen. Sie können auch Buckets identifizieren, die nicht den bewährten Datenschutzmethoden entsprechen, z. B. die Verwendung von S3 Replication oder S3 Versionierung. S3 Storage Lens analysiert Metriken, um kontextbezogene Empfehlungen zur Optimierung der Speicherkosten und zur Anwendung bewährter Datenschutzmethoden zu geben.

Sie können Amazon S3 Storage Lens verwenden, um Speichermetriken und Nutzungsdaten für alle AWS-Konten zu erfassen, die Teil Ihrer AWS Organizations-Hierarchie sind. Dafür müssen Sie AWS Organizations verwenden und den vertrauenswürdigen Zugriff von S3 Storage Lens über Ihr AWS Organizations-Verwaltungskonto aktivieren.

Nachdem Sie den vertrauenswürdigen Zugriff aktiviert haben, können Sie anderen Konten in Ihrer Organisation den delegierten Administratorzugriff gewähren. Diese Konten können dann S3-Storage-Lens-Konfigurationen und -Dashboards erstellen, die unternehmensweite Speichermetriken und Benutzerdaten sammeln.

Weitere Informationen zum Aktivieren des vertrauenswürdigen Zugriffs finden Sie unter Amazon S3 Storage Lens und AWS Organizations im AWS Organizations-Benutzerhandbuch.

Themen

Aktivieren des vertrauenswürdigen Zugriffs für S3 Storage Lens

Indem Sie den vertrauenswürdigen Zugriff aktivieren, ermöglichen Sie Amazon S3 Storage Lens, über die AWS Organizations-API-Operationen auf Ihre AWS Organizations-Hierarchie, -Mitgliedschaft und -Struktur zuzugreifen. S3 Storage Lens wird dann zu einem vertrauenswürdigen Service für die Struktur Ihrer gesamten Organisation.

Immer wenn eine Dashboard-Konfiguration definiert wird, erstellt S3 Storage Lens serviceverknüpfte Rollen in den Verwaltungskonten oder den delegierten Administratorkonten Ihrer Organisation. Die serviceverknüpfte Rolle erteilt S3 Storage Lens die Berechtigung für folgende Aufgaben:

- Beschreiben von Organisationen
- Auflisten von Konten
- Überprüfen einer AWS-Service-Zugriffsliste für die Organisationen
- Abrufen delegierter Administratoren für die Organisationen

S3 Storage Lens kann dann sicherstellen, dass es Zugriff hat, um die kontoübergreifende Metriken für die Konten in Ihren Organisationen zu sammeln. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für Amazon S3 Storage Lens.

Nachdem Sie den vertrauenswürdigen Zugriff aktiviert haben, können Sie anderen Konten in Ihrer Organisation den delegierten Administratorzugriff zuweisen. Wenn ein Konto als delegierter Administrator für einen Service gekennzeichnet ist, erhält es die Berechtigung, auf alle schreibgeschützten API-Operationen der Organisation zuzugreifen. Dieser Zugriff macht die Mitglieder und Strukturen Ihrer Organisation für den delegierten Administrator sichtbar, sodass dieser ebenfalls S3-Storage-Lens-Dashboards erstellen können.

> **Note**
>
> Der vertrauenswürdige Zugriff für Amazon S3 Storage Lens kann nur mit dem Verwaltungskonto aktiviert werden.

Deaktivieren des vertrauenswürdigen Zugriffs für S3 Storage Lens

Durch die Deaktivierung des vertrauenswürdigen Zugriffs beschränken Sie die S3-Storage-Lens-Funktionalität auf die Kontoebene. Darüber hinaus sieht jeder Kontoinhaber nur die Informationen von S3 Storage Lens für den Umfang seines Kontos und nicht für die gesamte Organisation. Alle Dashboards, die einen vertrauenswürdigen Zugriff erfordern, werden nicht mehr aktualisiert, behalten jedoch ihre historischen Daten für den Zeitraum bei, in dem Daten für Abfragen verfügbar sind.

> **Note**
>
> - Durch Deaktivieren des vertrauenswürdigen Zugriffs für S3 Storage Lens wird außerdem die Erfassung und Aggregation von Speichermetriken durch alle Dashboards auf Organisationsebene automatisch beendet.
> - Ihre Verwaltungs- und delegierten Administratorkonten können während des Zeitraums, in dem die Daten für Abfragen verfügbar sind, weiterhin die historischen Daten für Ihre bestehenden Dashboards auf Organisationsebene anzeigen.

Registrieren eines delegierten Administrators für S3 Storage Lens

Dashboards auf Organisationsebene können mithilfe des Verwaltungskontos Ihrer Organisation oder mit delegierten Administratorkonten erstellt werden. Der delegierte Administratorzugriff ermöglicht anderen Konten neben dem Verwaltungskonto, Dashboards auf Organisationsebene zu erstellen. Das Registrieren und Aufheben der Registrierung anderer Konten als delegierte Administratoren für eine Organisation kann lediglich über das Verwaltungskonto erfolgen.

Informationen zum Registrieren eines delegierten Administrators über die Amazon-S3-Konsole finden Sie unter Registrieren von delegierten Administratoren für S3 Storage Lens.

Sie können delegierte Administratoren im Verwaltungskonto auch über die AWS Organizations-REST-API, die AWS CLI oder SDKs registrieren. Weitere Informationen finden Sie unter RegisterDelegatedAdministrator in der AWS Organizations-API-Referenz.

> **Note**
>
> Bevor Sie mit der AWS Organizations-REST-API, der AWS CLI oder den SDKs delegierte Administratoren festlegen können, müssen Sie die Operation EnableAWSOrganizationsAccess aufrufen.

Aufheben der Registrierung eines delegierten Administrators für S3 Storage Lens

Sie können eine delegiertes Administratorkontos auch abmelden. Der delegierte Administratorzugriff ermöglicht anderen Konten neben dem Verwaltungskonto, Dashboards auf Organisationsebene zu erstellen. Konten, die als delegierte Administratoren festgelegt wurden, können nur über das Verwaltungskonto einer Organisation abgemeldet werden.

Informationen zum Abmelden eines delegierten Administrators über die S3-Konsole finden Sie unter Aufheben der Registrierung delegierter Administratoren für S3 Storage Lens.

Sie können delegierte Administratoren im Verwaltungskonto auch über die AWS Organizations-REST-API, die AWS CLI oder SDKs abmelden. Weitere Informationen finden Sie unter DeregisterDelegatedAdministrator in der AWS Organizations-API-Referenz.

> **Note**
>
> - Durch Abmelden eines delegierten Administrators wird die Aggregation neuer Speichermetriken durch alle Dashboards auf Organisationsebene, die von diesem delegierten Administrator erstellt wurden, automatisch beendet.
> - Die abgemeldeten delegierten Administratoren können weiterhin die historischen Daten für die Dashboards, die sie erstellt haben, anzeigen, während Daten für Abfragen verfügbar sind.

Berechtigungen für Amazon S3 Storage Lens

Amazon S3 Storage Lens erfordert neue Berechtigungen in AWS Identity and Access Management (IAM), um den Zugriff auf S3-Storage-Lens-Aktionen zu autorisieren. Zum Erteilen dieser Berechtigungen können Sie eine identitätsbasierte IAM-Richtlinie verwenden. Die Richtlinie lässt sich an IAM-Benutzer, -Gruppen oder -Rollen anhängen, wodurch diese die entsprechenden Berechtigungen erhalten. Diese Berechtigungen können die Fähigkeit beinhalten, S3 Storage Lens zu aktivieren oder zu deaktivieren oder auf ein beliebiges Dashboard oder eine Konfiguration von S3 Storage Lens zuzugreifen.

Der IAM-Benutzer oder die IAM-Rolle muss zu dem Konto gehören, das das Dashboard oder die Konfiguration erstellt hat oder besitzt, sofern nicht beide der folgenden Bedingungen zutreffen:

- Ihr Konto ist Mitglied von AWS Organizations.
- Sie haben über Ihr Verwaltungskonto als delegierter Administrator Zugriff auf die Erstellung von Dashboards auf Organisationsebene erhalten.

> **Note**
>
> - Sie können Dashboards von Amazon S3 Storage Lens nicht mithilfe der Stammbenutzer-Anmeldeinformationen Ihres Kontos aufrufen. Für den Zugriff auf S3-Storage-Lens-Dashboards müssen Sie die erforderlichen IAM-Berechtigungen einem neuen oder bestehenden IAM-Benutzer erteilen. Anschließend können Sie sich mit diesen Anmeldeinformationen anmelden, um auf S3-Storage-Lens-Dashboards zuzugreifen. Weitere Informationen finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.
> - Die Verwendung von S3 Storage Lens in der Amazon S3-Konsole kann mehrere Berechtigungen erfordern. Zum Bearbeiten eines Dashboards in der Konsole benötigen Sie beispielsweise die folgenden Berechtigungen:
>   - `s3:ListStorageLensConfigurations`
>   - `s3:GetStorageLensConfiguration`
>   - `s3:PutStorageLensConfiguration`

Themen

Festlegen von Kontoberechtigungen für die Verwendung von S3 Storage Lens

Zum Erstellen und Verwalten von S3-Storage-Lens-Dashboards und Storage-Lens-Dashboard-Konfigurationen benötigen Sie je nachdem, welche Aktionen Sie ausführen möchten, die folgenden Berechtigungen:

IAM-Berechtigungen für Amazon S3 Storage Lens

Action	IAM-Berechtigungen
Erstellen oder Aktualisieren eines S3-Storage-Lens-Dashboards in der Amazon S3-Konsole.	`s3:ListStorageLensConfigurations` `s3:GetStorageLensConfiguration`

Action	IAM-Berechtigungen
	`s3:GetStorageLensConfigurationTagging` `s3:PutStorageLensConfiguration` `s3:PutStorageLensConfigurationTagging`
Abrufen der Tags eines S3-Storage-Lens-Dashboards in der Amazon-S3-Konsole.	`s3:ListStorageLensConfigurations` `s3:GetStorageLensConfigurationTagging`
Ansehen eines S3-Storage-Lens-Dashboards in der Amazon S3-Konsole.	`s3:ListStorageLensConfigurations` `s3:GetStorageLensConfiguration` `s3:GetStorageLensDashboard`
Löschen eines S3-Storage-Lens-Dashboards in der Amazon S3-Konsole.	`s3:ListStorageLensConfigurations` `s3:GetStorageLensConfiguration` `s3:DeleteStorageLensConfiguration`
Erstellen oder Aktualisieren einer S3-Storage-Lens-Konfiguration mit der AWS CLI oder einem AWS SDK.	`s3:PutStorageLensConfiguration` `s3:PutStorageLensConfigurationTagging`
Abrufen der Tags einer S3-Storage-Lens-Konfiguration mit der AWS CLI oder einem AWS SDK.	`s3:GetStorageLensConfigurationTagging`
Ansehen einer S3-Storage-Lens-Konfiguration über die AWS CLI oder ein AWS SDK.	`s3:GetStorageLensConfiguration`

Action	IAM-Berechtigungen
Löschen einer S3-Storage-Lens-Konfiguration in der AWS CLI oder einem AWS SDK.	`s3:DeleteStorageLensConfiguration`

> **Note**
>
> - Sie können Ressourcen-Markierungen in einer IAM-Richtlinie verwenden, um Berechtigungen zu verwalten.
> - Ein IAM-Benutzer oder eine IAM-Rolle mit diesen Berechtigungen kann Metriken aus Buckets und Präfixen einsehen, für die der Benutzer bzw. die Rolle keine direkte Berechtigung zum Lesen oder Auflisten von Objekten hat.
> - Wenn bei S3-Storage-Lens-Dashboards mit aktivierten Metriken auf Präfixebene ein ausgewählter Präfixpfad mit einem Objektschlüssel übereinstimmt, zeigt das Dashboard den Objektschlüssel möglicherweise als anderes Präfix an.
> - Für Metrikexporte, die in einem Bucket in Ihrem Konto gespeichert werden, werden Berechtigungen mithilfe der bestehenden `s3:GetObject`-Berechtigung in der IAM-Richtlinie erteilt. In ähnlicher Weise kann das Verwaltungskonto oder die delegierten Administratorkonten einer Organisation für eine AWS Organizations-Entität IAM-Richtlinien verwenden, um Zugriffsberechtigungen für Dashboards und Konfigurationen auf Organisationsebene zu verwalten.

Festlegen von Kontoberechtigungen für die Verwendung von S3-Storage-Lens-Gruppen

Mithilfe von S3-Storage-Lens-Gruppen können Sie die Speicherverteilung innerhalb von Buckets anhand von Präfix, Suffix, Objekt-Tag, Objektgröße oder Objektalter nachvollziehen. Sie können Storage-Lens-Gruppe an Dashboards anhängen, um ihre aggregierten Metriken anzuzeigen.

Für die Verwendung von Storage-Lens-Gruppen benötigen Sie bestimmte Berechtigungen. Weitere Informationen finden Sie unter the section called “Berechtigungen für Storage-Lens-Gruppen”.

Festlegen von Berechtigungen zur Verwendung von S3 Storage Lens mit AWS Organizations

Sie können Amazon S3 Storage Lens verwenden, um Speichermetriken und Nutzungsdaten für alle Konten zu erfassen, die Teil Ihrer AWS Organizations-Hierarchie sind. Im Folgenden finden Sie die Aktionen und Berechtigungen für die Verwendung von S3 Storage Lens mit Organizations.

AWS Organizations-verwandte IAM-Berechtigungen für die Verwendung von S3 Storage Lens

Action	IAM-Berechtigungen
Aktivieren des vertrauenswürdigen Zugriffs für S3 Storage Lens für Ihre Organisation.	`organizations:EnableAWSServiceAccess`
Deaktivieren des vertrauenswürdigen Zugriffs für S3 Storage Lens für Ihre Organisation.	`organizations:DisableAWSServiceAccess`
Registrieren eines delegierten Administrators für die Erstellung von S3-Storage-Lens-Dashboards oder -Konfigurationen für Ihre Organisation.	`organizations:RegisterDelegatedAdministrator`
Abmelden eines delegierten Administrators, damit dieser keine S3 Storage Lens-Dashboards oder -Konfigurationen mehr für Ihre Organisation erstellen kann.	`organizations:DeregisterDelegatedAdministrator`
Zusätzliche Berechtigungen zum Erstellen von organisationsweiten S3-Storage-Lens-Konfigurationen	`organizations:DescribeOrganization` `organizations:ListAccounts` `organizations:ListAWSServiceAccessForOrganization` `organizations:ListDelegatedAdministrators` `iam:CreateServiceLinkedRole`

Anzeigen von Metriken mit Amazon S3 Storage Lens

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.

Standardmäßig sind alle Dashboards mit kostenlosen Metriken konfiguriert. Dazu gehören Metriken, anhand derer Sie die Nutzung und Aktivität in Ihrem S3-Speicher nachvollziehen, Ihre Speicherkosten optimieren und bewährte Methoden für Datenschutz und Zugriffsverwaltung implementieren können. Kostenlose Metriken werden bis auf Bucket-Ebene aggregiert. Bei kostenlosen Metriken stehen Daten bis zu 14 Tage lang für Abfragen zur Verfügung.

Zu den erweiterten Metriken und Empfehlungen gehören die folgenden zusätzlichen Funktionen, mit denen Sie weitere Einblicke in die Nutzung und Aktivität in Ihrem Speicher sowie bewährte Methoden zur Speicheroptimierung erhalten können:

- Kontextbezogene Empfehlungen (nur im Dashboard verfügbar)
- Erweiterte Metriken (einschließlich nach Buckets aggregierter Aktivitätsmetriken)
- Präfixzusammenfassung
- Aggregation von Storage-Lens-Gruppen
- Aggregation von Storage-Lens-Gruppen
- Amazon-CloudWatch-Veröffentlichung

Erweiterte Metrikdaten stehen 15 Monate für Abfragen zur Verfügung. Für die Verwendung von S3 Storage Lens mit fortschrittlichen Metriken fallen zusätzliche Gebühren an. Weitere Informationen finden Sie unter Amazon S3 – Preise. Weitere Informationen über kostenlose und erweiterte Metriken finden Sie unter Metrikauswahl.

Themen

- Anzeigen von S3-Storage-Lens-Metriken in den Dashboards
- Anzeigen von Amazon S3-Storage-Lens-Metriken mit einem Datenexport

- Überwachen von Metriken von S3 Storage Lens in CloudWatch

Anzeigen von S3-Storage-Lens-Metriken in den Dashboards

In der Amazon-S3-Konsole bietet S3 Storage Lens ein interaktives Standard-Dashboard, mit dem Sie Erkenntnisse zu Daten sowie entsprechende Trends visualisieren können. Zudem lässt sich das Dashboard verwenden, um Ausreißer zu kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten sowie zur Anwendung bewährter Datenschutzmethoden einzusehen. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Konto-, Bucket-, AWS-Region-, Präfix- oder Storage-Lens-Gruppenebene zu generieren. Wenn Sie S3 Storage Lens für AWS Organizations aktiviert haben, besteht zudem die Möglichkeit, Erkenntnisse auf Organisationsebene zu generieren (z. B. Daten für alle Konten, die der AWS Organizations-Hierarchie angehören). Das Dashboard wird immer für das letzte Datum geladen, für das Metriken verfügbar sind.

Das Standard-Dashboard von S3 Storage Lens in der Konsole heißt default-account-dashboard. Gemäß Amazon-S3-Vorkonfiguration werden in diesem Dashboard die zusammengefassten Erkenntnisse und Trends Ihres gesamten Kontos visualisiert und täglich in der S3-Konsole aktualisiert. Sie können den Konfigurationsbereich des Standard-Dashboards nicht ändern, aber Sie haben die Möglichkeit, statt der kostenlosen Metriken die kostenpflichtigen erweiterten Metriken und Empfehlungen auszuwählen. Mit erweiterten Metriken und Empfehlungen können Sie auf zusätzliche Metriken und Funktionen zugreifen. Zu diesen Funktionen gehören erweiterte Metrikkategorien, Aggregation auf Präfixebene, kontextbezogene Empfehlungen und Amazon-CloudWatch-Veröffentlichung.

Sie können das Standard-Dashboard deaktivieren, aber nicht löschen. Wenn Sie Ihr Standard-Dashboard deaktivieren, wird es nicht mehr aktualisiert. Sie erhalten keine neuen täglichen Metriken mehr in S3 Storage Lens oder im Konto-Snapshot auf der Seite Buckets. Sie können im Standard-Dashboard weiterhin historische Daten anzeigen, bis der Zeitraum von 14 Tagen für Datenabfragen abläuft. Wenn Sie erweiterte Metriken und Empfehlungen aktiviert haben, beträgt dieser Zeitraum 15 Monate. Sie können das Standard-Dashboard innerhalb des Ablaufzeitraums wieder aktivieren, um auf diese Daten zuzugreifen.

Sie können zusätzliche S3-Storage-Lens-Dashboards erstellen und ihren Umfang nach AWS-Regionen, S3-Buckets oder Konten festlegen. Zudem sind Sie in der Lage, Dashboards auch nach Organisation einzugrenzen, wenn Sie Storage Lens für AWS Organizations aktiviert haben. Wenn Sie ein S3-Storage-Lens-Dashboard erstellen oder bearbeiten, definieren Sie den Dashboard-Umfang und die Metrikauswahl.

Sie können zusätzliche Dashboards, die Sie erstellen, deaktivieren oder löschen.

- Wenn Sie ein Dashboard deaktivieren, wird es nicht mehr aktualisiert und Sie erhalten keine neuen täglichen Metriken mehr. Sie können historische Daten für kostenlose Metriken dann noch bis zum Ende des Ablaufzeitraums von 14 Tagen einsehen. Wenn Sie erweiterte Metriken und Empfehlungen für das Dashboard aktiviert haben, beträgt dieser Zeitraum 15 Monate. Sie können das Dashboard innerhalb des Ablaufzeitraums wieder aktivieren, um auf diese Daten zuzugreifen.
- Wenn Sie ein Dashboard löschen, gehen alle Dashboard-Konfigurationseinstellungen verloren. Sie erhalten dann keine neuen täglichen Metriken mehr und können nicht mehr auf die mit dem Dashboard verbundenen historischen Daten zugreifen. Wenn Sie auf die historischen Daten eines gelöschten Dashboards zugreifen möchten, müssen Sie ein neues Dashboard mit demselben Namen in derselben Heimatregion erstellen.

Themen

Anzeigen eines Amazon S3-Storage-Lens-Dashboards

Das folgende Verfahren zeigt, wie Sie ein S3-Storage-Lens-Dashboard in der S3-Konsole ansehen können. Anwendungsfallbasierte Anleitungen, die zeigen, wie Sie das Dashboard verwenden können, um Ihre Kosten zu optimieren, bewährte Verfahren zu implementieren und die Leistung von Anwendungen zu verbessern, die auf S3-Buckets zugreifen, finden Sie unter Anwendungsfälle für Metriken von Amazon S3 Storage Lens.

> **Note**
>
> Sie können Dashboards von Amazon S3 Storage Lens nicht mithilfe der Root–Benutzer-Anmeldeinformationen Ihres Kontos aufrufen. Für den Zugriff auf S3-Storage-Lens-Dashboards müssen Sie die erforderlichen AWS Identity and Access Management (IAM)-Berechtigungen einem neuen oder bestehenden IAM-Benutzer erteilen. Anschließend können Sie sich mit diesen Anmeldeinformationen anmelden, um auf S3-Storage-Lens-Dashboards zuzugreifen. Weitere Informationen finden Sie unter Berechtigungen für Amazon S3 Storage Lens und Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.

   Ihr Dashboard wird in S3 Storage Lens geöffnet. Der Abschnitt Snapshot for date (Snapshot für Datum) zeigt das letzte Datum an, an dem S3 Storage Lens Metriken gesammelt hat. Ihr Dashboard wird immer für das letzte Datum geladen, für das Metriken verfügbar sind.
4. (Optional) Wenn Sie das Datum für Ihr S3-Storage-Lens-Dashboard ändern möchten, wählen Sie in der oberen rechten Datumsauswahl ein neues Datum aus.
5. (Optional) Gehen Sie wie folgt vor, wenn Sie temporäre Filter anwenden möchten, mit denen Sie den Umfang Ihrer Dashboard-Daten weiter einschränken können:

   a. Erweitern Sie den Bereich Filter.

   b. Wenn Sie nach bestimmten Konten, AWS-Regionen, Speicherklassen, Buckets, Präfixen oder Storage-Lens-Gruppen filtern möchten, wählen Sie die Optionen aus, nach denen gefiltert werden soll.

      > **Note**
      >
      > Der Filter Präfixe und der Filter Storage-Lens-Gruppen können nicht gleichzeitig angewendet werden.

   c. Wenn Sie einen Filter aktualisieren möchten, wählen Sie Apply (Anwenden) aus.

   d. Zum Entfernen eines Filters klicken Sie auf das X neben dem Filter.
6. Wählen Sie zum Anzeigen von Daten für eine bestimmte Metrik in einem beliebigen Abschnitt Ihres S3-Storage-Lens-Dashboards im Feld Metric (Metrik) den Namen der entsprechenden Metrik aus.
7. In jedem Diagramm oder jeder Visualisierung des S3-Storage-Lens-Dashboards können Sie mithilfe der Registerkarten Konto, AWS-Regionen, Speicherklassen, Buckets, Präfixe oder Storage-Lens-Gruppen tiefere Aggregationsebenen aufrufen. Ein Beispiel finden Sie unter Entdecken Sie kalte Amazon S3-Buckets.

Grundlegendes zum S3-Storage-Lens-Dashboard

Ihr S3-Storage-Lens-Dashboard enthält eine primäre Registerkarte Overview (Übersicht) und bis zu fünf zusätzlichen Registerkarten für die einzelnen Aggregationsebenen:

- Konten
- AWS-Regionen
- Speicherklassen
- Buckets
- Präfixe
- Storage-Lens-Gruppen

Auf der Registerkarte Overview (Übersicht) werden Ihre Dashboard-Daten in drei verschiedene Abschnitte zusammengefasst: Snapshot for date (Snapshot für Datum), Trends and distributions (Trends und Verteilungen) und Top N overview (Top N-Übersicht).

Weitere Informationen zur Verwendung Ihres S3-Storage-Lens-Dashboards finden Sie in den folgenden Abschnitten.

Snapshot

Der Abschnitt Snapshot for date (Snapshot für Datum) zeigt zusammenfassende Metriken an, die S3 Storage Lens für das ausgewählte Datum gesammelt hat. Diese zusammenfassenden Metriken umfassen die folgenden Metriken:

- Gesamtspeicher – Die Gesamtmenge des verwendeten Speichers in Byte
- Anzahl der Objekte – Die Gesamtanzahl der Objekte im AWS-Konto
- Durchschnittliche Objektgröße – Die durchschnittliche Objektgröße
- Aktive Buckets – Die Gesamtanzahl der aktiven Buckets in aktiver Nutzung mit Speicher > 0 Byte im Konto
- Konten – Die Anzahl der Konten, deren Speicher zum Umfang gehören. Dieser Wert ist 1, es sei denn, Sie verwenden AWS Organizations und S3 Storage Lens hat vertrauenswürdigen Zugriff mit einer gültigen serviceverknüpften Rolle. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für Amazon S3 Storage Lens.
- Buckets – Die Gesamtanzahl der Buckets im Konto

Metrik-Daten

Für jede Metrik, die im Snapshot angezeigt wird, können Sie die folgenden Daten sehen:

- Metrikname – Der Name der Metrik
- Metrikkategorie – Die Kategorie, in der die Metrik organisiert ist
- Gesamtsumme für Datum – Die Gesamtzahl für das ausgewählte Datum
- % Änderung – Die prozentuale Änderung seit dem letzten Snapshot-Datum
- 30-Tage-Trend – Eine Trendlinie, die die Änderungen der Metrik über einen Zeitraum von 30 Tagen zeigt
- Empfehlung – Eine kontextbezogene Empfehlung, die auf den Daten basiert, die im Snapshot bereitgestellt werden. Empfehlungen sind mit erweiterten Metriken und Empfehlungen verfügbar. Weitere Informationen finden Sie unter Empfehlungen.

Metrikkategorien

Sie können optional Ihren Dashboard-Abschnitt Snapshot for date (Snapshot für Datum) aktualisieren, um Metriken für andere Kategorien anzuzeigen. Wenn Sie Snapshot-Daten für zusätzliche Metriken sehen möchten, können Sie aus den folgenden Metrics categories (Metrikkategorien) wählen:

- Kostenoptimierung
- Datenschutz
- Aktivität (mit erweiterten Metriken verfügbar)
- Zugriffsverwaltung
- Leistung
- Ereignisse

Im Abschnitt Snapshot for date (Snapshot für Datum) wird nur eine Auswahl von Metriken für jede Kategorie angezeigt. Um alle Metriken für eine bestimmte Kategorie anzuzeigen, wählen Sie die Metrik in den Abschnitten Trends and distributions (Trends und Verteilungen) oder Top N overview (Top N-Übersicht) aus. Weitere Informationen zu den Metrikkategorien finden Sie unter Metrikkategorien. Eine vollständige Liste der Metriken von S3 Storage Lens finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Trends und Verteilungen

Der zweite Abschnitt der Registerkarte Overview (Übersicht) lautet Trends and distributions (Trends und Verteilungen). Im Abschnitt Trends and distributions (Trends und Verteilungen) können Sie zwei Metriken auswählen, um sie über einen von Ihnen definierten Zeitraum zu vergleichen. Der Abschnitt Trends and distributions (Trends und Verteilungen) zeigt die Beziehung zwischen zwei Metriken im Zeitverlauf. Dieser Abschnitt enthält Diagramme, mit denen Sie die Verteilung von Storage class (Speicherklasse) und Region zwischen den beiden Trends sehen können, die Sie beobachten. Sie können optional einen Datenpunkt in einem der Diagramme aufschlüsseln, um ihn eingehender zu analysieren.

Eine Anleitung, die den Abschnitt Trends and distribution (Trends und Verteilungen) verwendet, finden Sie unter Identifizieren von Buckets, die keine serverseitige Verschlüsselung mit AWS KMS für die Standardverschlüsselung (SSE-KMS) verwenden.

Top N-Übersicht

Der dritte Abschnitt des S3-Storage-Lens-Dashboards ist die Top-N-Übersicht (in aufsteigender oder absteigender Reihenfolge sortiert). In diesem Abschnitt werden die ausgewählten Metriken in den Top-N-Konten, -AWS-Regionen, -Buckets, -Präfixen oder Storage-Lens-Gruppen angezeigt. Wenn Sie S3 Storage Lens für AWS Organizations aktiviert haben, können Sie die ausgewählten Metriken für die Organisation einsehen.

Eine Anleitung, die den Abschnitt Top N overview (Top N-Übersicht) verwendet, finden Sie unter Identifizieren Sie Ihre größten S3-Buckets.

Aufschlüsselung und Analyse nach Optionen

Um für eine reibungslose Analyse zu sorgen, bietet das S3-Storage-Lens-Dashboard ein Menü mit Aktionen, das bei Auswahl eines Diagrammwerts angezeigt wird. Wenn Sie dieses Menü verwenden möchten, wählen Sie einen beliebigen Diagrammwert aus, um die zugehörigen Metrikwerte zu sehen. Es stehen zwei Optionen zur Auswahl:

- Mit der Aktion Drill down (Aufgliedern) wird der ausgewählte Wert als Filter auf alle Registerkarten Ihres Dashboards angewendet. Sie können dann einen Drilldown für eine ausführlichere Analyse durchführen.
- Bei Auswahl der Aktion Analysieren nach wird die ausgewählte Registerkarte Dimension im Dashboard geöffnet und der Wert wird als Filter angewendet. Zu diesen Registerkarten gehören Konten, AWS-Regionen, Speicherklassen, Buckets, Präfixe (für Dashboards mit

aktivierten Optionen Erweiterte Metriken und Präfixaggregation) sowie Storage-Lens-Gruppen (für Dashboards mit aktivierten Optionen Erweiterte Metriken und Aggregation von Storage-Lens-Gruppen). Mit Analysieren nach lassen sich die Daten im Kontext der neuen Dimension anzeigen und noch ausführlicher analysieren.

Die Aktionen Aufgliedern und Analysieren nach sind möglicherweise deaktiviert, wenn das Ergebnis unlogische Ergebnisse liefern würde oder keinen Wert hätte. Beide Aktionen Aufgliedern und Analysieren nach führen dazu, dass Filter auf alle Registerkarten des Dashboards angewendet werden (zusätzlich zu den vorhandenen Filtern). Bei Bedarf lassen sich Filter auch entfernen.

Registerkarten

Die Registerkarten auf Dimensionsebene bieten eine detaillierte Ansicht aller Werte innerhalb einer bestimmten Dimension. Die Registerkarte AWS-Regionen zeigt beispielsweise Metriken für alle AWS-Regionen an und die Registerkarte Buckets Metriken für alle Buckets. Jeder Dimensions-Tab hat das gleiche Layout, das aus vier Abschnitten besteht:

- Im Trenddiagramm werden Ihre Top-N-Elemente innerhalb der Dimension in den letzten 30 Tagen für die ausgewählte Metrik angezeigt. Standardmäßig enthält dieses Diagramm die Top 10 Elemente. Sie können die Anzahl auf minimal 3 Elemente verringern oder auf maximal 50 Elemente erhöhen.
- Das Histogramm enthält ein vertikales Balkendiagramm für die ausgewählte Kombination aus Datum und Metrik. Wenn Sie eine große Anzahl von Elementen in diesem Diagramm anzeigen möchten, müssen Sie ggf. horizontal scrollen.
- Im Blasen-Analysediagramm werden alle Elemente innerhalb der Dimension dargestellt. In diesem Diagramm wird die erste Metrik auf der X-Achse und die zweite Metrik auf der Y-Achse dargestellt. Die dritte Metrik wird durch die Größe der Blase repräsentiert.
- Die metrische Rasteransicht enthält jedes Element in der Dimension. Die Werte werden in Zeilen aufgeführt. Die Spalten stellen jede verfügbare Metrik dar, die zur einfacheren Navigation nach Tabs mit Metrikkategorien angeordnet sind.

Anzeigen von Amazon S3-Storage-Lens-Metriken mit einem Datenexport

Amazon S3-Storage-Lens-Metriken werden täglich als Metrik-Exportdateien im CSV- oder Apache Parquet-Format generiert und in einem S3-Bucket in Ihrem Konto abgelegt. Aus diesem Bucket können Sie den Metrik-Export in die Analysetools Ihrer Wahl aufnehmen, z. B. Amazon QuickSight und Amazon Athena, wo Sie Speichernutzungs- und Aktivitätstrends analysieren können.

Themen

Verwenden eines AWS KMS key zur Verschlüsselung von Metrik-Exporten

Um Amazon S3 Storage Lens die Berechtigung zum Verschlüsseln Ihrer Metrikexporte mit einem kundenverwalteten Schlüssel zu erteilen, müssen Sie eine Schlüsselrichtlinie verwenden. Gehen Sie folgendermaßen vor, um Ihre Schlüsselrichtlinie zu aktualisieren, damit Sie S3-Storage-Lens-Metrik-Exporte mit einem KMS-Schlüssel verschlüsseln können.

So gewähren Sie S3 Storage Lens Berechtigungen zum Verschlüsseln mithilfe Ihres KMS-Schlüssels

1. Melden Sie sich bei der AWS Management Console mit dem AWS-Konto an, dem der kundenverwaltete Schlüssel gehört.
2. Öffnen Sie die AWS KMS-Konsole unter https://console.aws.amazon.com/kms.
3. Um die AWS-Region zu ändern, verwenden Sie die Regionsauswahl in der oberen rechten Ecke der Seite.
4. Klicken Sie im linken Navigationsbereich auf Customer managed keys (Vom Kunden verwaltete Schlüssel).
5. Wählen Sie unter Kundenverwaltete Schlüssel den Schlüssel aus, den Sie zum Verschlüsseln der Metrikexporte verwenden möchten. AWS KMS keys sind regionsspezifisch und müssen sich in derselben Region wie der S3-Ziel-Bucket für den Metrikexport befinden.
6. Wählen Sie unter Key policy (Schlüsselrichtlinie) die Option Switch to policy view (Zur Richtlinienansicht wechseln) aus.
7. Um die Schlüsselrichtlinie zu aktualisieren, wählen Sie Edit (Bearbeiten).
8. Fügen Sie unter Edit key policy (Schlüsselrichtlinie bearbeiten) die folgende Schlüsselrichtlinie zu der vorhandenen Schlüsselrichtlinie hinzu. Wenn Sie diese Richtlinie verwenden möchten, ersetzen Sie *`user input placeholders`* durch eigene Informationen.

```
{
    "Sid": "Allow Amazon S3 Storage Lens use of the KMS key",
     "Effect": "Allow",
    "Principal": {
```

```
            "Service": "storage-lens.s3.amazonaws.com"
        },
        "Action": [
            "kms:GenerateDataKey"
        ],
        "Resource": "*",
        "Condition": {
           "StringEquals": {
               "aws:SourceArn": "arn:aws:s3:us-east-1:source-account-id:storage-
lens/your-dashboard-name",
               "aws:SourceAccount": "source-account-id"
            }
         }
}
```

9. Wählen Sie Save Changes.

Weitere Informationen zum Erstellen von kundenverwalteten Schlüsseln und zum Verwenden von Schlüsselrichtlinien finden Sie unter den folgenden Themen im AWS Key Management Service Entwicklerhandbuch:

- Erste Schritte
- Verwenden von Schlüsselrichtlinien in AWS KMS

Sie können auch die API Operation AWS KMS-Schlüsselrichtlinie PUT (PutKeyPolicy) verwenden, um die Schlüsselrichtlinie in die kundenverwalteten Schlüssel zu kopieren, die Sie zur Verschlüsselung der Metrik-Exporte mithilfe der REST-API, der AWS CLI oder SDKs verwenden möchten.

Was ist ein S3-Storage-Lens-Exportmanifest?

Angesichts der großen Menge von aggregierten Daten kann der tägliche Metrik-Export von S3 Storage Lens in mehrere Dateien aufgeteilt werden. Die Manifestdatei `manifest.json` beschreibt, wo sich die Metrik-Exportdateien für den jeweiligen Tag befinden. Immer wenn ein neuer Export bereitgestellt wird, wird ein neues zugehöriges Manifest erstellt. Jedes in der `manifest.json`-Datei enthaltene Manifest bietet Metadaten und andere grundlegende Informationen zum Export.

Die Manifestinformationen umfassen die folgenden Eigenschaften:

- `sourceAccountId` – Die Konto-ID des Konfigurationsinhabers.
- `configId` – Eine eindeutige ID für das Dashboard.

- `destinationBucket` – Der Amazon-Ressourcenname (ARN) des Ziel-Buckets, in dem der Metrik-Export abgelegt wird.
- `reportVersion` – Die Version des Exports.
- `reportDate` – Das Datum des Berichts.
- `reportFormat` – Das Format des Berichts.
- `reportSchema` – Das Schema des Berichts.
- `reportFiles` – Die tatsächliche Liste der Exportberichtsdateien, die sich im Ziel-Bucket befinden.

Nachfolgend sehen Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen CSV-formatierten Export.

```
{
   "sourceAccountId":"123456789012",
   "configId":"my-dashboard-configuration-id",
   "destinationBucket":"arn:aws:s3:::destination-bucket",
   "reportVersion":"V_1",
   "reportDate":"2020-11-03",
   "reportFormat":"CSV",

 "reportSchema":"version_number,configuration_id,report_date,aws_account_number,aws_region,stor
   "reportFiles":[
        {
            "key":"DestinationPrefix/StorageLens/123456789012/my-dashboard-
configuration-id/V_1/reports/dt=2020-11-03/a38f6bc4-2e3d-4355-ac8a-e2fdcf3de158.csv",
            "size":1603959,
            "md5Checksum":"2177e775870def72b8d84febe1ad3574"
        }
      ]
}
```

Nachfolgend sehen Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen Export im Parquet-Format.

```
{
   "sourceAccountId":"123456789012",
   "configId":"my-dashboard-configuration-id",
   "destinationBucket":"arn:aws:s3:::destination-bucket",
   "reportVersion":"V_1",
```

```
   "reportDate":"2020-11-03",
   "reportFormat":"Parquet",
   "reportSchema":"message s3.storage.lens { required string version_number;
 required string configuration_id; required string report_date; required string
 aws_account_number; required string aws_region; required string storage_class;
 required string record_type; required string record_value; required string
 bucket_name; required string metric_name; required long metric_value; }",
   "reportFiles":[
      {
         "key":"DestinationPrefix/StorageLens/123456789012/my-dashboard-configuration-
id/V_1/reports/dt=2020-11-03/bd23de7c-b46a-4cf4-bcc5-b21aac5be0f5.par",
         "size":14714,
         "md5Checksum":"b5c741ee0251cd99b90b3e8eff50b944"
      }
}
```

Sie können den Metrik-Export so konfigurieren, dass er als Teil Ihrer Dashboard-Konfiguration in der Amazon S3-Konsole oder mithilfe der Amazon S3-REST-API, der AWS CLI oder SDKs generiert wird.

Grundlegendes zum Amazon S3-Storage-Lens-Exportschema

Die folgende Tabelle enthält das Schema Ihres S3-Storage-Lens-Metrik-Exports.

Attributname	Datentyp	Spaltenname	Beschreibung
`VersionNumber`	Zeichenfolge	`version_number`	Die Version der verwendeten S3-Storage-Lens-Metriken.
`ConfigurationId`	Zeichenfolge	`configuration_id`	Die `configuration_id` Ihrer S3-Storage-Lens-Konfiguration.
`ReportDate`	Zeichenfolge	`report_date`	Das Datum, an dem die Metriken erfasst wurden.

Attributname	Datentyp	Spaltenname	Beschreibung
`AwsAccountNumber`	Zeichenfolge	`aws_account_number`	Ihre AWS-Konto-Nummer.
`AwsRegion`	Zeichenfolge	`aws_region`	Die AWS-Region, für die die Metriken erfasst werden.
`StorageClass`	Zeichenfolge	`storage_class`	Die Speicherklasse des betreffenden Buckets.
`RecordType`	ENUM	`record_type`	Der Typ des gemeldeten Artefakts (ACCOUNT, BUCKET oder PREFIX).
`RecordValue`	Zeichenfolge	`record_value`	Der Wert des `RecordType` - Artefakts. **Note** Der `record_value` ist URL-kodiert.
`BucketName`	Zeichenfolge	`bucket_name`	Der Name des Buckets, der gemeldet wird.
`MetricName`	Zeichenfolge	`metric_name`	Der Name der Metrik, die gemeldet wird.
`MetricValue`	Long	`metric_value`	Der Wert der Metrik, die gemeldet wird.

Beispiel für einen S3-Storage-Lens-Metrik-Export

Im Folgenden sehen Sie ein Beispiel für einen auf diesem Schema basierenden S3-Storage-Lens-Metrik-Export.

> **Note**
>
> Sie können Metriken für Storage-Lens-Gruppen ermitteln, indem Sie in der Spalte `record_type` nach dem Wert `STORAGE_LENS_GROUP_BUCKET` oder `STORAGE_LENS_GROUP_ACCOUNT` suchen. In der Spalte `record_value` wird der Amazon-Ressourcenname (ARN) für die Storage-Lens-Gruppe angezeigt (z. B. `arn:aws:s3:us-east-1:123456789012:storage-lens-group/slg-1`).

version_r	configuration_id	report_date	aws_account_number	aws_region	storage_class	record_type	record_value	bucket_name	metric_name	metric_value
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			StorageBytes	2478830621
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ObjectCount	1598962
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ReplicatedStorageBytes	20000
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ReplicatedObjectCount	20
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			EncryptedStorageBytes	2478828742
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			EncryptedObjectCount	1598961
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			DeleteMarkerObjectCount	1500
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ObjectLockEnabledStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ObjectLockEnabledObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			CurrentVersionStorageBytes	2478830621
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			CurrentVersionObjectCount	1598962
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			NonCurrentVersionStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			NonCurrentVersionObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			IncompleteMultipartUploadStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			IncompleteMultipartUploadObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	StorageBytes	29996800
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	ObjectCount	12264
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	ReplicatedStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	ReplicatedObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	EncryptedStorageBytes	29996800
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	EncryptedObjectCount	12264
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	DeleteMarkerObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	ObjectLockEnabledStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	ObjectLockEnabledObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	CurrentVersionStorageBytes	29996800
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	CurrentVersionObjectCount	12264
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	NonCurrentVersionStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	NonCurrentVersionObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	IncompleteMultipartUploadStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fu	cloudtrail-log-sf	IncompleteMultipartUploadObjectCount	0

Im Folgenden finden Sie ein Beispiel für einen S3-Storage-Lens-Metrikexport mit Storage-Lens-Gruppendaten.

version_numbe	configuration_id	report_date	aws_account_num	aws_region	storage_class	record_type	record_value	bucket_name	metric_name	metric_value
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		StorageBytes	3128548856
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ObjectCount	4440
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		EncryptedStorageBytes	3128548856
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		EncryptedObjectCount	4440
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		DeleteMarkerObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ObjectLockEnabledStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ObjectLockEnabledObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		CurrentVersionStorageBytes	3128548856
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		CurrentVersionObjectCount	4440
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		NonCurrentVersionStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		NonCurrentVersionObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMultipartUploadStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMultipartUploadObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		DeleteMarkerStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedStorageBytesSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedObjectCountSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMPUStorageBytesOlderThan7Days	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMPUObjectCountOlderThan7Days	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	StorageBytes	676863200
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ObjectCount	3000
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	EncryptedStorageBytes	676863200
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	EncryptedObjectCount	3000
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	DeleteMarkerObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ObjectLockEnabledStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ObjectLockEnabledObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	CurrentVersionStorageBytes	676863200
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	CurrentVersionObjectCount	3000
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	NonCurrentVersionStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	NonCurrentVersionObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMultipartUploadStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMultipartUploadObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	DeleteMarkerStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedStorageBytesSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedObjectCountSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMPUStorageBytesOlderThan7Days	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMPUObjectCountOlderThan7Days	0

Überwachen von Metriken von S3 Storage Lens in CloudWatch

Sie können S3-Storage-Lens-Metriken in Amazon CloudWatch veröffentlichen, um eine einheitliche Ansicht Ihres Betriebszustands in CloudWatch-Dashboards zu erstellen. Sie können auch CloudWatch-Funktionen wie Alarme und ausgelöste Aktionen, Metrikmathematik und Anomalieerkennung verwenden, um S3-Storage-Lens-Metriken zu überwachen und Maßnahmen zu ergreifen. Darüber hinaus ermöglichen CloudWatch-API-Operationen Anwendungen, einschließlich Drittanbietern, den Zugriff auf Ihre S3-Storage-Lens-Metriken. Weitere Informationen zu CloudWatch finden Sie im Amazon-CloudWatch-Benutzerhandbuch.

Sie können die CloudWatch-Veröffentlichungsoption für neue oder vorhandene Dashboard-Konfigurationen mithilfe der Amazon-S3-Konsole, der Amazon-S3-REST-API, AWS CLI und AWS SDKs aktivieren. Dashboards, die auf fortschrittliche Metriken und Empfehlungen von S3 Storage Lens aktualisiert werden, können die CloudWatch-Veröffentlichungsoption verwenden. Informationen zu Preisen für erweiterte Metriken und Empfehlungen von S3 Storage Lens finden Sie unter Amazon-S3-Preise. Es fallen keine zusätzlichen Gebühren für die Veröffentlichung von CloudWatch-Metriken an. Es gelten jedoch andere CloudWatch-Gebühren z. B. für Dashboards, Alarme und API-Aufrufe. Weitere Informationen hierzu finden Sie unter Amazon CloudWatch – Preise.

Die Metriken von S3 Storage Lens werden in CloudWatch in dem Konto veröffentlicht, das die S3-Storage-Lens-Konfiguration besitzt. Nachdem Sie die CloudWatch-Veröffentlichungsoption für erweiterte Metriken und Empfehlungen aktiviert haben, können Sie in CloudWatch auf Metriken auf

Organisations-, Konto- und Bucket-Ebene zugreifen. Metriken auf Präfixebene sind in CloudWatch nicht verfügbar.

> **Note**
>
> S3-Storage-Lens-Metriken sind tägliche Metriken und werden einmal täglich in CloudWatch veröffentlicht. Wenn Sie S3-Storage-Lens-Metriken in CloudWatch abfragen, muss der Zeitraum für die Abfrage 1 Tag (86400 Sekunden) betragen. Nachdem Ihre täglichen S3-Storage-Lens-Metriken in Ihrem S3–Storage-Lens-Dashboard in der Amazon-S3-Konsole angezeigt werden, kann es ein paar Stunden dauern, bis diese Metriken in CloudWatch angezeigt werden. Wenn Sie die CloudWatch-Veröffentlichungsoption für S3-Storage-Lens-Metriken zum ersten Mal aktivieren, kann es bis zu 24 Stunden dauern, bis Ihre Metriken auf CloudWatch veröffentlicht werden.

Nachdem Sie die CloudWatch-Veröffentlichungsoption aktiviert haben, können Sie die folgenden CloudWatch-Funktionen verwenden, um Ihre Daten in S3 Storage Lens zu überwachen und zu analysieren:

- Dashboards – Erstellen Sie mithilfe von CloudWatch-Dashboards benutzerdefinierte Dashboards für S3 Storage Lens. Teilen Sie Ihr CloudWatch-Dashboard mit Personen, die keinen direkten Zugriff auf Ihr AWS-Konto haben, teamübergreifend, mit Stakeholdern und mit Personen außerhalb Ihrer Organisation.
- Alarme und ausgelöste Aktionen – Konfigurieren Sie Alarme, die Metriken überwachen und Maßnahmen ergreifen, wenn ein Schwellenwert überschritten wird. Sie können beispielsweise einen Alarm konfigurieren, der eine Amazon-SNS-Benachrichtigung sendet, wenn die Metrik Incomplete Multipart Upload Bytes (Unvollständige Bytes für mehrteilige Uploads) an drei aufeinanderfolgenden Tagen 1 GB überschreitet.
- Anomalieerkennung – Ermöglichen Sie die Anomalieerkennung, um Metriken kontinuierlich zu analysieren, normale Baselines und Oberflächenanomalien zu ermitteln. Sie können einen Anomalieerkennungsalarm basierend auf dem erwarteten Wert einer Metrik erstellen. Sie können beispielsweise Anomalien für die Metrik Object Lock Enabled Bytes (Bytes mit aktivierter Objektsperre), um das unbefugte Entfernen von Objektsperreinstellungen zu erkennen.
- Mithilfe von Metrikberechnungen können Sie mehrere -Metriken mit S3-Storage-Lens-Metriken abfragen und mathematische Ausdrücke verwenden, um neue Zeitreihen basierend auf diesen Metriken zu erstellen. Sie können beispielsweise eine neue Metrik erstellen, um die

durchschnittliche Objektgröße zu erhalten, indem Sie `StorageBytes` durch `ObjectCount` dividieren.

Weitere Informationen zur CloudWatch-Veröffentlichungsoption für S3-Storage-Lens-Metriken finden Sie in den folgenden Themen.

Themen

- Metriken und Dimensionen von S3 Storage Lens
- Aktivieren der CloudWatch-Veröffentlichung für S3 Storage Lens
- Arbeiten mit Metriken von S3 Storage Lens in CloudWatch

Metriken und Dimensionen von S3 Storage Lens

Um S3-Storage-Lens-Metriken an CloudWatch zu senden, müssen Sie die CloudWatch-Veröffentlichungsoption in den fortschrittlichen Metriken und Empfehlungen von S3 Storage Lens aktivieren. Sobald erweiterte Metriken aktiviert sind, können Sie CloudWatch-Dashboards verwenden, um S3-Storage-Lens-Metriken zusammen mit anderen Anwendungsmetriken zu überwachen und eine einheitliche Ansicht auf Ihren Betriebszustand zu erstellen. Sie können Dimensionen verwenden, um Ihre S3-Storage-Lens-Metriken in CloudWatch nach Organisation, Konto, Bucket, Speicherklasse, Region und Kennzahlen-Konfigurations-ID zu filtern.

Die Metriken von S3 Storage Lens werden in CloudWatch in dem Konto veröffentlicht, das die S3-Storage-Lens-Konfiguration besitzt. Nachdem Sie die CloudWatch-Veröffentlichungsoption für erweiterte Metriken und Empfehlungen aktiviert haben, können Sie in CloudWatch auf Metriken auf Organisations-, Konto- und Bucket-Ebene zugreifen. Metriken auf Präfixebene sind in CloudWatch nicht verfügbar.

> **Note**
>
> S3-Storage-Lens-Metriken sind tägliche Metriken und werden einmal täglich in CloudWatch veröffentlicht. Wenn Sie S3-Storage-Lens-Metriken in CloudWatch abfragen, muss der Zeitraum für die Abfrage 1 Tag (86400 Sekunden) betragen. Nachdem Ihre täglichen S3-Storage-Lens-Metriken in Ihrem S3–Storage-Lens-Dashboard in der Amazon-S3-Konsole angezeigt werden, kann es ein paar Stunden dauern, bis diese Metriken in CloudWatch angezeigt werden. Wenn Sie die CloudWatch-Veröffentlichungsoption für S3-Storage-Lens-

Metriken zum ersten Mal aktivieren, kann es bis zu 24 Stunden dauern, bis Ihre Metriken auf CloudWatch veröffentlicht werden.

Weitere Informationen zu S3-Storage-Lens-Metriken und -Dimensionen in CloudWatch finden Sie in den folgenden Themen.

Themen

- Metriken
- Dimensionen

Metriken

In CloudWatch Metriken von S3 Storage Lens als Metriken zur Verfügung. S3-Storage-Lens-Metriken werden auf dem `AWS/S3/Storage-Lens`-Namespace veröffentlicht. Dieser Namespace ist nur für S3-Storage-Lens-Metriken vorgesehen. Amazon-S3-Bucket, Anforderungs- und Replikations-Metriken werden im `AWS/S3`-Namespace veröffentlicht.

Die Metriken von S3 Storage Lens werden in CloudWatch in dem Konto veröffentlicht, das die S3-Storage-Lens-Konfiguration besitzt. Nachdem Sie die CloudWatch-Veröffentlichungsoption für erweiterte Metriken und Empfehlungen aktiviert haben, können Sie in CloudWatch auf Metriken auf Organisations-, Konto- und Bucket-Ebene zugreifen. Metriken auf Präfixebene sind in CloudWatch nicht verfügbar.

In S3 Storage Lens werden Metriken aggregiert und nur in der dafür vorgesehenen Heimatregion gespeichert. Die Metriken von S3 Storage Lens werden auch in CloudWatch in der Heimatregion veröffentlicht, die Sie in der S3-Storage-Lens-Konfiguration angeben.

Eine vollständige Liste der S3-Storage-Lens-Metriken, einschließlich einer Liste der in CloudWatch verfügbaren Metriken, finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Note

Die gültige Statistik für S3-Storage-Lens-Metriken in CloudWatch ist Durchschnittlich. Weitere Informationen zu Statistiken in CloudWatch finden Sie unter CloudWatch-Statistikdefinitionen im Amazon-CloudWatch-Benutzerhandbuch.

Die Granularität von S3-Storage-Lens-Metriken in CloudWatch

S3 Storage Lens bietet Metriken für Organisation, Konto, Bucket und Präfixgranularität. S3 Storage Lens veröffentlicht S3-Storage-Lens-Metriken auf Organisations-, Konto- und Bucketebene auf CloudWatch. In CloudWatch sind S3-Storage-Lens-Metriken auf Präfixebene nicht verfügbar.

Weitere Informationen zur Granularität der in CloudWatch verfügbaren S3-Storage-Lens-Metriken finden Sie in der folgenden Liste:

- Organisation – In den Mitgliedskonten Ihrer Organisation aggregierte Metriken. S3 Storage Lens veröffentlicht Metriken für Mitgliedskonten in CloudWatch im Verwaltungskonto.
  - Organisation und Konto – Metriken für die Mitgliedskonten Ihrer Organisation.
  - Organisation und Bucket – Metriken für Amazon-S3-Buckets in den Mitgliedskonten Ihrer Organisation.
- Konto (Nicht-Organisationsebene) – Metriken, die in den Buckets Ihres Kontos zusammengefasst sind.
- Bucket (Nicht-Organisationsebene) – Metriken für einen bestimmten Bucket. In CloudWatch veröffentlicht S3 Storage Lens diese Metriken in dem AWS-Konto, das die S3-Storage-Lens-Konfiguration erstellt hat. S3 Storage Lens veröffentlicht diese Metriken nur für nicht organisatorische Konfigurationen.

Dimensionen

Wenn S3 Storage Lens Daten an CloudWatch sendet, werden jeder Metrik Dimensionen angefügt. Dimensionen sind Kategorien, die die Eigenschaften von Metriken beschreiben. Sie können Dimensionen verwenden, um die von CloudWatch zurückgegebenen Ergebnisse zu filtern.

Beispielsweise haben alle S3-Storage-Lens-Metriken in CloudWatch die `configuration_id`-Dimension. Sie können diese Dimension verwenden, um zwischen Metriken zu unterscheiden, die mit einer bestimmten S3-Storage-Lens-Konfiguration verknüpft sind. Die `organization_id` identifiziert Metriken auf Organisationsebene. Weitere Informationen zu Dimensionen der CloudWatch finden Sie unter Dimensionen im CloudWatch-Entwicklerhandbuch.

Abhängig von der Granularität der Metriken sind für S3-Storage-Lens-Metriken unterschiedliche Dimensionen verfügbar. Sie können beispielsweise die `organization_id`-Dimension zum Filtern von Metriken auf Organisationsebene nach der AWS Organizations-ID verwenden. Sie können diese Dimension jedoch nicht für Metriken auf Bucket- und Kontoebene verwenden. Weitere Informationen finden Sie unter Metriken mit Dimensionen filtern.

Auf der folgenden Tabelle können Sie erkennen, welche Dimensionen für Ihre S3-Storage-Lens-Konfiguration verfügbar sind.

Dimension	Beschreibung	Bucket	Konto	Organisation	Organisation und Konto	Organisation und Bucket
`configuration_id`	Der in den Metriken gemeldete Dashboard-Name für die S3-Storage-Lens-Konfiguration	.	.	.	.	.
`metrics_version`	Die Version der S3-Storage-Lens-Metriken. Die Metrikversion hat einen festen Wert von `1.0`.	.	.	.	.	.
`organization_id`	Die AWS Organizations-ID der Metriken	.	.	.	.	.
`aws_account_number`	Das AWS-Konto, das den Metriken zugeordnet ist	.	.	.	.	.
`aws_region`	Die AWS-Region für die Metriken	.	.	.	.	.
`bucket_name`	Der Name des in den Metriken gemeldeten S3-Buckets	.	.	.	.	.
`storage_class`	Die Speicherklasse für den Bucket, die in den Metriken gemeldet wird	.	.	.	.	.
`record_type`	Die Granularität der Metriken: ORGANIZATION, ACCOUNT, BUCKET	BUCKET	-	-	-	ORGANIZATION

Aktivieren der CloudWatch-Veröffentlichung für S3 Storage Lens

Sie können S3-Storage-Lens-Metriken in Amazon CloudWatch veröffentlichen, um eine einheitliche Ansicht Ihres Betriebszustands in CloudWatch-Dashboards zu erstellen. Sie können

auch CloudWatch-Funktionen wie Alarme und ausgelöste Aktionen, Metrikmathematik und Anomalieerkennung verwenden, um S3-Storage-Lens-Metriken zu überwachen und Maßnahmen zu ergreifen. Darüber hinaus ermöglichen CloudWatch-API-Operationen Anwendungen, einschließlich Drittanbietern, den Zugriff auf Ihre S3-Storage-Lens-Metriken. Weitere Informationen zu CloudWatch finden Sie im Amazon-CloudWatch-Benutzerhandbuch.

Die Metriken von S3 Storage Lens werden in CloudWatch in dem Konto veröffentlicht, das die S3-Storage-Lens-Konfiguration besitzt. Nachdem Sie die CloudWatch-Veröffentlichungsoption für erweiterte Metriken und Empfehlungen aktiviert haben, können Sie in CloudWatch auf Metriken auf Organisations-, Konto- und Bucket-Ebene zugreifen. Metriken auf Präfixebene sind in CloudWatch nicht verfügbar.

Sie können CloudWatch-Unterstützung für neue oder vorhandene Dashboard-Konfigurationen mithilfe der S3-Konsole, Amazon-S3-REST-APIs, AWS CLI und AWS SDKs aktivieren. Die CloudWatch-Veröffentlichungsoption ist für Dashboards verfügbar, die auf erweiterte Metriken und Empfehlungen von S3 Storage Lens aktualisiert wurden. Informationen zu Preisen für erweiterte Metriken und Empfehlungen von S3 Storage Lens finden Sie unter Amazon-S3-Preise. Es fallen keine zusätzlichen Gebühren für die Veröffentlichung von CloudWatch-Metriken an. Es gelten jedoch andere CloudWatch-Gebühren z. B. für Dashboards, Alarme und API-Aufrufe.

Weitere Informationen zur Aktivierung der CloudWatch-Veröffentlichungsoption für S3-Storage-Lens-Metriken finden Sie in den folgenden Themen.

> **Note**
>
> S3-Storage-Lens-Metriken sind tägliche Metriken und werden einmal täglich in CloudWatch veröffentlicht. Wenn Sie S3-Storage-Lens-Metriken in CloudWatch abfragen, muss der Zeitraum für die Abfrage 1 Tag (86400 Sekunden) betragen. Nachdem Ihre täglichen S3-Storage-Lens-Metriken in Ihrem S3–Storage-Lens-Dashboard in der Amazon-S3-Konsole angezeigt werden, kann es ein paar Stunden dauern, bis diese Metriken in CloudWatch angezeigt werden. Wenn Sie die CloudWatch-Veröffentlichungsoption für S3-Storage-Lens-Metriken zum ersten Mal aktivieren, kann es bis zu 24 Stunden dauern, bis Ihre Metriken auf CloudWatch veröffentlicht werden.
> Derzeit können S3-Storage-Lens-Metriken nicht über CloudWatch-Streams verwendet werden.

Verwenden der S3-Konsole

Wenn Sie ein S3-Storage-Lens-Dashboard aktualisieren, können Sie den Dashboard-Namen oder die Heimatregion nicht ändern. Sie können auch den Umfang des Standard-Dashboards nicht ändern, das auf den gesamten Speicher Ihres Kontos abgestimmt ist.

So aktualisieren Sie das S3-Storage-Lens-Dashboard, um die CloudWatch-Veröffentlichung zu aktivieren

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich S3 Storage Lens und dann Dashboards aus.
3. Wählen Sie das Dashboard aus, das Sie bearbeiten möchten, und wählen Sie dann Bearbeiten.
4. Unter Metrikauswahl, wählen Sie Fortschrittliche Metriken und Empfehlungen aus.

   Fortschrittliche Metriken und Empfehlungen sind gegen Aufpreis erhältlich. Zu den erweiterten Metriken und Empfehlungen gehören ein Zeitraum von 15 Monaten für Datenabfragen, auf Präfixebene aggregierte Nutzungsmetriken, nach Bucket aggregierte Aktivitätsmetriken, die CloudWatch-Veröffentlichungsoption und kontextbezogene Empfehlungen, die Ihnen helfen, Speicherkosten zu optimieren und bewährte Methoden für den Datenschutz anzuwenden. Weitere Informationen finden Sie unter Amazon S3 – Preise.
5. Wählen Sie unter Fortschrittliche Metriken und Empfehlungsfunktionen auswählen die Option CloudWatch-Veröffentlichung aus.

   > **Important**
   >
   > Wenn Ihre Konfiguration die Präfix-Aggregation für Nutzungsmetriken aktiviert, werden Metriken auf Präfixebene nicht in CloudWatch veröffentlicht. Auf CloudWatch werden nur S3-Storage-Lens-Metriken auf Bucket-, Konto- und Organisationsebene veröffentlicht.
6. Wählen Sie Save Changes.

So erstellen Sie ein neues S3-Storage-Lens-Dashboard, das CloudWatch-Unterstützung ermöglicht

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.

3. Klicken Sie auf Dashboard erstellen.
4. Definieren Sie unter General (Allgemein) die folgenden Konfigurationsoptionen:

   a. Geben Sie im Feld Dashboard name (Dashboard-Name) Ihren Dashboard-Namen ein.

      Dashboard-Namen müssen weniger als 65 Zeichen lang sein und dürfen keine Sonderzeichen oder Leerzeichen enthalten. Sie können den Dashboard-Namen nicht mehr ändern, nachdem Sie Ihr Dashboard erstellt haben.

   b. Wählen Sie die Heimatregion für Ihr Dashboard aus.

      Metriken für alle Regionen, die in diesem Dashboard-Umfang enthalten sind, werden zentral in der angegebenen Heimatregion gespeichert. In CloudWatch sind Metriken für S3 Storage Lens auch in der Heimatregion verfügbar. Sie können die Heimatregion nicht ändern, nachdem Sie Ihr Dashboard erstellt haben.

5. (Optional) Sie fügen ein Tag hinzu, indem Sie Add tag (Tag hinzufügen) auswählen und den Schlüssel und den Wert für den Tag eingeben.

   > **Note**
   >
   > Sie können Ihrer Dashboard-Konfiguration bis zu 50 Markierungen hinzufügen.

6. Definieren Sie den Umfang für Ihre Konfiguration:

   a. Wenn Sie eine Konfiguration auf Organisationsebene erstellen, wählen Sie die Konten aus, die in die Konfiguration aufgenommen werden sollen: Include all accounts in your configuration (Alle Konten in die Konfiguration einschließen) oder Limit the scope to your signed-in account (Umfang auf das angemeldete Konto beschränken).

      > **Note**
      >
      > Wenn Sie eine Konfiguration auf Organisationsebene erstellen, die alle Konten enthält, können Sie nur Regionen und keine Buckets ein- oder ausschließen.

   b. Wählen Sie die Regionen und Buckets aus, die S3 Storage Lens im Dashboard einschließen soll, indem Sie die folgenden Schritte ausführen:

      - Um alle Regionen einzuschließen, wählen Sie Regionen und Buckets einschließen aus.

- Um bestimmte Regionen einzubeziehen, löschen Sie Alle Regionen einschließen. Unter Wählen Sie die einzuschließenden Regionen wählen Sie die Regionen aus, die S3 Storage Lens im Dashboard enthalten soll.
- Um bestimmte Buckets einzubeziehen, löschen Sie Alle Buckets einschließen. Wählen Sie unter Einzuschließende Buckets auswählen die Buckets aus, die S3 Storage Lens in das Dashboard aufnehmen soll.

> **Note**
>
> Sie können bis zu 50 Buckets auswählen.

7. Unter Metrics selection (Metrikauswahl) wählen Sie Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) aus.

   Weitere Informationen zu fortschrittlichen Metriken und Preisempfehlungen finden Sie unter Amazon-S3-Preise.

8. Wählen Sie unter Advanced metrics and recommendations features (Erweiterte Metrik- und Empfehlungsfunktionen) die Optionen aus, Sie aktivieren möchten:

   - Advanced metrics (Erweiterte Metriken)
   - CloudWatch-Veröffentlichung

   > **Important**
   >
   > Wenn Sie die Präfix-Aggregation für Ihre S3-Storage-Lens-Konfiguration aktivieren, werden Metriken auf Präfixebene nicht in CloudWatch veröffentlicht. Auf CloudWatch werden nur S3-Storage-Lens-Metriken auf Bucket-, Konto- und Organisationsebene veröffentlicht.

   - Präfixzusammenfassung

   > **Note**
   >
   > Weitere Informationen über erweiterte Metriken und Empfehlungen finden Sie unter Metrikauswahl.

9. Wenn Sie Advanced Metrics (Erweiterte Metriken) aktiviert haben, wählen Sie die Advanced metrics categories (Erweiterte Metrikkategorien) aus, die Sie in Ihrem S3-Storage-Lens-Dashboard anzeigen möchten:

   - Metriken für Aktivitäten
   - Detailed status code metrics (Detaillierte Statuscode-Metriken)
   - Advanced cost optimization metrics (Erweiterte Kostenoptimierungsmetriken)
   - Advanced data protection metrics (Erweiterte Datensicherheitsmetriken)

   Weitere Informationen zu den Metrikkategorien finden Sie unter Metrikkategorien. Eine vollständige Liste der Metriken finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.
10. (Optional) Konfigurieren Sie den Metrikexport.

    Weitere Informationen zum Konfigurieren eines Metrikexports finden Sie unter Schritt Erstellen eines Amazon S3-Storage-Lens-Dashboards.
11. Klicken Sie auf Dashboard erstellen.

Verwendung von AWS CLI

Das folgende AWS CLI-Beispiel aktiviert die CloudWatch-Veröffentlichungsoption mithilfe einer Konfiguration mit erweiterten Metriken und Empfehlungen auf Organisationsebene von S3 Storage Lens. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control put-storage-lens-configuration --account-id=555555555555 --config-
id=your-configuration-id --region=us-east-1 --storage-lens-configuration=file://./
config.json

config.json
{
  "Id": "SampleS3StorageLensConfiguration", //Use this property to identify your S3
 Storage Lens configuration.
  "AwsOrg": { //Use this property when enabling S3 Storage Lens for AWS Organizations.
    "Arn": "arn:aws:organizations::123456789012:organization/o-abcdefgh"
  },
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
```

```
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "DetailedStatusCodesMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "ActivityMetrics": {
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "AdvancedCostOptimizationMetrics": {
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "DetailedStatusCodesMetrics": {
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "PrefixLevel":{
        "StorageMetrics":{
          "IsEnabled":true, //Mark this as false if you want only free metrics.
          "SelectionCriteria":{
            "MaxDepth":5,
            "MinStorageBytesPercentage":1.25,
            "Delimiter":"/"
          }
        }
      }
    }
  },
  "Exclude": { //Replace with "Include" if you prefer to include Regions.
    "Regions": [
      "eu-west-1"
    ],
    "Buckets": [ //This attribute is not supported for AWS Organizations-level
configurations.
      "arn:aws:s3:::source_bucket1"
    ]
  },
  "IsEnabled": true, //Whether the configuration is enabled
```

```
  "DataExport": { //Details about the metrics export
    "S3BucketDestination": {
      "OutputSchemaVersion": "V_1",
      "Format": "CSV", //You can add "Parquet" if you prefer.
      "AccountId": "111122223333",
      "Arn": "arn:aws:s3:::destination-bucket-name", // The destination bucket for your
 metrics export must be in the same Region as your S3 Storage Lens configuration.
      "Prefix": "prefix-for-your-export-destination",
      "Encryption": {
        "SSES3": {}
      }
    },
    "CloudWatchMetrics": {
      "IsEnabled": true //Mark this as false if you want to export only free metrics.
    }
  }
}
```

Verwenden des AWS-SDK für Java

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.ActivityMetrics;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.CloudWatchMetrics;
import com.amazonaws.services.s3control.model.Format;
import com.amazonaws.services.s3control.model.Include;
import com.amazonaws.services.s3control.model.OutputSchemaVersion;
import com.amazonaws.services.s3control.model.PrefixLevel;
import com.amazonaws.services.s3control.model.PrefixLevelStorageMetrics;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.S3BucketDestination;
import com.amazonaws.services.s3control.model.SSES3;
import com.amazonaws.services.s3control.model.SelectionCriteria;
import com.amazonaws.services.s3control.model.StorageLensAwsOrg;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensDataExport;
```

```
import com.amazonaws.services.s3control.model.StorageLensDataExportEncryption;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateAndUpdateDashboard {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        String exportAccountId = "Destination Account ID";
        String exportBucketArn = "arn:aws:s3:::destBucketName"; // The destination
 bucket for your metrics export must be in the same Region as your S3 Storage Lens
 configuration.
        String awsOrgARN = "arn:aws:organizations::123456789012:organization/o-
abcdefgh";
        Format exportFormat = Format.CSV;

        try {
            SelectionCriteria selectionCriteria = new SelectionCriteria()
                    .withDelimiter("/")
                    .withMaxDepth(5)
                    .withMinStorageBytesPercentage(10.0);
            PrefixLevelStorageMetrics prefixStorageMetrics = new
 PrefixLevelStorageMetrics()
                    .withIsEnabled(true)
                    .withSelectionCriteria(selectionCriteria);
            BucketLevel bucketLevel = new BucketLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withAdvancedCostOptimizationMetrics(new
 AdvancedCostOptimizationMetrics().withIsEnabled(true))
                    .withAdvancedDataProtectionMetrics(new
 AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
 DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withPrefixLevel(new
 PrefixLevel().withStorageMetrics(prefixStorageMetrics));
            AccountLevel accountLevel = new AccountLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withAdvancedCostOptimizationMetrics(new
 AdvancedCostOptimizationMetrics().withIsEnabled(true))
```

```
                    .withAdvancedDataProtectionMetrics(new
AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withBucketLevel(bucketLevel);

            Include include = new Include()
                    .withBuckets(Arrays.asList("arn:aws:s3:::bucketName"))
                    .withRegions(Arrays.asList("us-west-2"));

            StorageLensDataExportEncryption exportEncryption = new
StorageLensDataExportEncryption()
                    .withSSES3(new SSES3());
            S3BucketDestination s3BucketDestination = new S3BucketDestination()
                    .withAccountId(exportAccountId)
                    .withArn(exportBucketArn)
                    .withEncryption(exportEncryption)
                    .withFormat(exportFormat)
                    .withOutputSchemaVersion(OutputSchemaVersion.V_1)
                    .withPrefix("Prefix");
            CloudWatchMetrics cloudWatchMetrics = new CloudWatchMetrics()
                    .withIsEnabled(true);
            StorageLensDataExport dataExport = new StorageLensDataExport()
                    .withCloudWatchMetrics(cloudWatchMetrics)
                    .withS3BucketDestination(s3BucketDestination);

            StorageLensAwsOrg awsOrg = new StorageLensAwsOrg()
                    .withArn(awsOrgARN);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withInclude(include)
                    .withDataExport(dataExport)
                    .withAwsOrg(awsOrg)
                    .withIsEnabled(true);

            List<StorageLensTag> tags = Arrays.asList(
                    new StorageLensTag().withKey("key-1").withValue("value-1"),
                    new StorageLensTag().withKey("key-2").withValue("value-2")
            );

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
```

```
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
                    .withTags(tags)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwenden der REST-API

Wenn Sie die CloudWatch-Veröffentlichungsoption mit der Amazon-S3-REST-API aktivieren möchten, können Sie PutStorageLensConfiguration verwenden.

Nächste Schritte

Nachdem Sie die CloudWatch-Veröffentlichungsoption aktiviert haben, können Sie in CloudWatch auf Ihre S3-Storage-Lens-Metriken zugreifen. Sie können CloudWatch-Funktionen auch nutzen, um Ihre S3-Storage-Lens-Daten in CloudWatch zu überwachen und zu analysieren. Weitere Informationen finden Sie unter den folgenden Themen:

- Metriken und Dimensionen von S3 Storage Lens
- Arbeiten mit Metriken von S3 Storage Lens in CloudWatch

Arbeiten mit Metriken von S3 Storage Lens in CloudWatch

Sie können S3-Storage-Lens-Metriken in Amazon CloudWatch veröffentlichen, um eine einheitliche Ansicht Ihres Betriebszustands in CloudWatch-Dashboards zu erstellen. Sie können auch CloudWatch-Funktionen wie Alarme und ausgelöste Aktionen, Metrikmathematik und

Anomalieerkennung verwenden, um S3-Storage-Lens-Metriken zu überwachen und Maßnahmen zu ergreifen. Darüber hinaus ermöglichen CloudWatch-API-Operationen Anwendungen, einschließlich Drittanbietern, den Zugriff auf Ihre S3-Storage-Lens-Metriken. Weitere Informationen zu CloudWatch finden Sie im Amazon-CloudWatch-Benutzerhandbuch.

Sie können die CloudWatch-Veröffentlichungsoption für neue oder vorhandene Dashboard-Konfigurationen mithilfe der Amazon-S3-Konsole, der Amazon-S3-REST-APIs, AWS CLI und AWS SDKs aktivieren. Die CloudWatch-Veröffentlichungsoption ist für Dashboards verfügbar, die auf erweiterte Metriken und Empfehlungen von S3 Storage Lens aktualisiert wurden. Informationen zu Preisen für erweiterte Metriken und Empfehlungen von S3 Storage Lens finden Sie unter Amazon-S3-Preise. Es fallen keine zusätzlichen Gebühren für die Veröffentlichung von CloudWatch-Metriken an. Es gelten jedoch andere CloudWatch-Gebühren z. B. für Dashboards, Alarme und API-Aufrufe. Weitere Informationen hierzu finden Sie unter Amazon CloudWatch – Preise.

Die Metriken von S3 Storage Lens werden in CloudWatch in dem Konto veröffentlicht, das die S3-Storage-Lens-Konfiguration besitzt. Nachdem Sie die CloudWatch-Veröffentlichungsoption für erweiterte Metriken und Empfehlungen aktiviert haben, können Sie in CloudWatch auf Metriken auf Organisations-, Konto- und Bucket-Ebene zugreifen. Metriken auf Präfixebene sind in CloudWatch nicht verfügbar.

> **Note**
>
> S3-Storage-Lens-Metriken sind tägliche Metriken und werden einmal täglich in CloudWatch veröffentlicht. Wenn Sie S3-Storage-Lens-Metriken in CloudWatch abfragen, muss der Zeitraum für die Abfrage 1 Tag (86400 Sekunden) betragen. Nachdem Ihre täglichen S3-Storage-Lens-Metriken in Ihrem S3–Storage-Lens-Dashboard in der Amazon-S3-Konsole angezeigt werden, kann es ein paar Stunden dauern, bis diese Metriken in CloudWatch angezeigt werden. Wenn Sie die CloudWatch-Veröffentlichungsoption für S3-Storage-Lens-Metriken zum ersten Mal aktivieren, kann es bis zu 24 Stunden dauern, bis Ihre Metriken auf CloudWatch veröffentlicht werden.
> Derzeit können S3-Storage-Lens-Metriken nicht über CloudWatch-Streams verwendet werden.

Weitere Informationen zum Arbeiten mit S3-Storage-Lens-Metriken in CloudWatch finden Sie in den folgenden Themen.

Themen

- Arbeiten mit CloudWatch-Dashboards
- Alarme einstellen, Aktionen auslösen und Anomalieerkennung verwenden
- Metriken mit Dimensionen filtern
- Berechnen neuer Metriken mit Metrikmathematik
- Verwenden von Suchausdrücken in Diagrammen

Arbeiten mit CloudWatch-Dashboards

Sie können Cloudwatch-Dashboards verwenden, um S3-Storage-Lens-Metriken zusammen mit anderen Anwendungsmetriken zu überwachen und eine einheitliche Sicht auf Ihren Betriebszustand zu erstellen. Dashboards sind eine anpassbare Homepage in der CloudWatch-Konsole, die Sie verwenden können, um Ihre -Ressourcen in einer zentralen Ansicht zu überwachen.

CloudWatch verfügt über eine breite Berechtigungssteuerung, die die Beschränkung des Zugriffs auf bestimmte Metriken oder Dimensionen nicht unterstützt. Benutzer in Ihrem Konto oder Ihrer Organisation, die Zugriff auf CloudWatch haben, haben Zugriff auf Metriken für alle S3-Storage-Lens-Konfigurationen, bei denen die CloudWatch-Supportoption aktiviert ist. Sie können Berechtigungen für bestimmte Dashboards nicht wie in S3 Storage Lens verwalten. Weitere Informationen zu CloudWatch-Berechtigungen finden Sie unter Verwalten von Zugriffsberechtigungen für Ihre CloudWatch-Ressourcen im Amazon-CloudWatch-Benutzerhandbuch.

Weitere Informationen zur Verwendung von CloudWatch-Dashboards und zum Konfigurieren von Berechtigungen finden Sie unter Verwenden von Amazon-CloudWatch-Dashboards und Freigeben von CloudWatch-Dashboards im Amazon-CloudWatch-Benutzerhandbuch.

Alarme einstellen, Aktionen auslösen und Anomalieerkennung verwenden

Sie können CloudWatch-Alarme konfigurieren, die S3-Storage-Lens-Metriken in CloudWatch ansehen, und Maßnahmen ergreifen, wenn ein Schwellenwert überschritten wird. Sie können beispielsweise einen Alarm konfigurieren, der eine Amazon-SNS-Benachrichtigung sendet, wenn die Metrik Incomplete Multipart Upload Bytes (Unvollständige Bytes für mehrteilige Uploads) an drei aufeinanderfolgenden Tagen 1 GB überschreitet.

Sie können auch die Anomalieerkennung aktivieren, um Ihre S3-Storage-Lens-Metriken kontinuierlich zu analysieren und normale Baselines und Oberflächenanomalien zu ermitteln. Sie können Anomalieerkennungsalarme basierend auf dem erwarteten Wert einer Metrik erstellen. Sie können beispielsweise Anomalien für die Metrik Object Lock Enabled Bytes (Bytes mit aktivierter Objektsperre), um das unbefugte Entfernen von Objektsperreinstellungen zu erkennen.

Weitere Informationen und Beispiele finden Sie unter Verwenden von Amazon-CloudWatch-Alarmen und Erstellen eines Alarms aus einer Metrik in einem Diagramm im Amazon-CloudWatch-Benutzerhandbuch.

Metriken mit Dimensionen filtern

Sie können Dimensionen verwenden, um S3-Storage-Lens-Metriken in der CloudWatch-Konsole zu filtern. Beispielsweise können Sie nach `configuration_id`, `aws_account_number`, `aws_region`, `bucket_name` und mehr filtern.

S3 Storage Lens unterstützt mehrere Dashboard-Konfigurationen pro Konto. Dies bedeutet, dass verschiedene Konfigurationen denselben Bucket enthalten können. Wenn diese Metriken in CloudWatch veröffentlicht werden, enthält der Bucket doppelte Metriken in CloudWatch. Um Metriken nur für eine bestimmte S3-Storage-Lens-Konfiguration in CloudWatch anzuzeigen, können Sie die `configuration_id`-Dimension verwenden. Wenn Sie nach `configuration_id` filtern, sehen Sie nur Metriken, die mit der Konfiguration verknüpft sind, die Sie identifizieren.

Weitere Informationen zum Filtern nach Konfigurations-ID finden Sie unter Suche nach verfügbaren Metriken im Amazon-CloudWatch-Benutzerhandbuch.

Berechnen neuer Metriken mit Metrikmathematik

Mithilfe von Metrikberechnungen können Sie mehrere Metriken mit S3 Storage Lens abfragen und mathematische Ausdrücke verwenden, um neue Zeitreihen basierend auf diesen Metriken zu erstellen. Sie können beispielsweise eine neue Metrik für unverschlüsselte Objekte erstellen, indem Sie verschlüsselte Objekte von der Objektanzahl subtrahieren. Sie können auch eine Metrik erstellen, um die durchschnittliche Objektgröße zu erhalten, indem Sie `StorageBytes` durch `ObjectCount` oder die Anzahl der Bytes dividieren, auf die an einem Tag zugegriffen wird, indem Sie `BytesDownloaded` durch `StorageBytes` dividieren.

Weitere Informationen finden Sie unter Metrikmathematik verwenden im Amazon-CloudWatch-Benutzerhandbuch.

Verwenden von Suchausdrücken in Diagrammen

Mit S3-Storage-Lens-Metriken können Sie einen Suchausdruck erstellen. Sie können beispielsweise einen Suchausdruck für alle Metriken namens IncompleteMultipartUploadStorageBytes erstellen und dem Ausdruck `SUM` hinzufügen. Mit diesem Suchausdruck können Sie Ihre gesamten unvollständigen mehrteiligen Upload-Bytes über alle Dimensionen Ihres Speichers in einer einzigen Metrik sehen.

Dieses Beispiel zeigt die Syntax, die Sie verwenden würden, um einen Suchausdruck für alle Metriken mit dem Namen IncompleteMultipartUploadStorageBytes zu erstellen.

```
SUM(SEARCH('{AWS/S3/Storage-
Lens,aws_account_number,aws_region,configuration_id,metrics_version,record_type,storage_class}
 MetricName="IncompleteMultipartUploadStorageBytes"', 'Average',86400))
```

Weitere Informationen zu dieser Syntax finden Sie unter Syntax für CloudWatch-Suchausdrücke im Amazon-CloudWatch-Benutzerhandbuch. Informationen zum Erstellen eines CloudWatch-Diagramms mit einem Suchausdruck finden Sie unter Erstellen eines CloudWatch-Diagramms mit einem Suchausdruck im Amazon-CloudWatch-Benutzerhandbuch.

Anwendungsfälle für Metriken von Amazon S3 Storage Lens

Sie können Ihr Dashboard von Amazon S3 Storage Lens verwenden, um Erkenntnisse und Trends zu visualisieren, Ausreißer zu markieren und Empfehlungen einzusehen. Die Metriken von S3 Storage Lens sind in Kategorien unterteilt, die den wichtigsten Anwendungsfällen entsprechen. Diese Metriken bieten Ihnen folgende Möglichkeiten:

- Identifizieren von Möglichkeiten zur Kostenoptimierung
- Anwenden von bewährten Datenschutzmethoden
- Anwenden von bewährten Zugriffsverwaltungsmethoden
- Verbessern der Leistung von Anwendungs-Workloads

Mithilfe der Metriken zur Kostenoptimierung können Sie Möglichkeiten ermitteln, wie Sie Ihre Amazon-S3-Speicherkosten senken können. Sie können Buckets mit mehrteiligen Uploads identifizieren, die älter als 7 Tage sind, oder Buckets, in denen sich nicht aktuelle Versionen ansammeln.

In ähnlicher Weise können Sie Datenschutzmetriken verwenden, um Buckets zu identifizieren, die nicht den bewährten Datenschutzmethoden Ihres Unternehmens entsprechen. Sie können beispielsweise Buckets identifizieren, die keine AWS Key Management Service-Schlüssel (SSE-KMS) für die Standardverschlüsselung verwenden oder für die S3 Versioning nicht aktiviert ist.

Mit den Zugriffsverwaltungsmetriken von S3 Storage Lens können Sie Bucket-Einstellungen für S3 Object Ownership identifizieren, sodass Sie Zugriffssteuerungslisten (ACLs) Berechtigungen zu Bucket-Richtlinien migrieren und ACLs deaktivieren können.

Wenn Sie die Option S3 Storage Lens advanced metrics (Erweiterte Metriken von S3 Storage Lens) aktiviert haben, können Sie die detaillierten Statuscode-Metriken verwenden, um die Anzahl von erfolgreichen oder fehlgeschlagenen Anforderungen zu erhalten. Anhand dieser Informationen können Sie Zugriffs- oder Leistungsprobleme beheben.

Mithilfe erweiterter Metriken können Sie auch auf zusätzliche Metriken zur Kostenoptimierung und zum Datenschutz zugreifen, anhand derer Sie Möglichkeiten zur weiteren Senkung Ihrer Gesamtkosten für den S3-Speicher identifizieren und besser an bewährte Methoden für den Schutz Ihrer Daten anpassen können. Erweiterte Metriken zur Kostenoptimierung umfassen beispielsweise die Anzahl von Lebenszyklusregeln, mit der Sie Buckets identifizieren können, für die keine Lebenszyklusregeln festgelegt sind, um unvollständige mehrteilige Uploads, die älter als 7 Tage sind, ablaufen zu lassen. Zu den erweiterten Datenschutzmetriken gehört die Anzahl der Replikationsregeln.

Weitere Informationen zu den Metrikkategorien finden Sie unter Metrikkategorien. Eine vollständige Liste der Metriken von S3 Storage Lens finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Themen

- Verwenden von Amazon S3 Storage Lens zur Optimierung Ihrer Speicherkosten
- Verwenden von S3 Storage Lens zum Schutz Ihrer Daten
- Verwenden von S3 Storage Lens zur Überwachung der Einstellungen für die Objekteigentümerschaft
- Verwenden der Metriken von S3 Storage Lens zur Verbesserung der Leistung

Verwenden von Amazon S3 Storage Lens zur Optimierung Ihrer Speicherkosten

Sie können die Kostenoptimierungsmetriken von S3 Storage Lens verwenden, um die Gesamtkosten Ihres S3-Speichers zu senken. Mithilfe von Metriken zur Kostenoptimierung können Sie überprüfen, dass Sie Amazon S3 kostengünstig und gemäß den bewährten Methoden konfiguriert haben. Sie können beispielsweise die folgenden Möglichkeiten zur Kostenoptimierung identifizieren:

- Buckets mit unvollständigen mehrteiligen Uploads, die älter als 7 Tage sind
- Buckets, in denen sich zahlreiche nicht aktuelle Versionen ansammeln
- Buckets, die keine Lebenszyklusregeln zum Abbruch unvollständiger mehrteiliger Uploads haben
- Buckets, die keine Lebenszyklusregeln für das Ablaufen von Objekten mit nicht aktuellen Versionen haben

- Buckets, die keine Lebenszyklusregeln für den Übergang von Objekten in eine andere Speicherklasse haben

Sie können diese Daten dann verwenden, um Ihren Buckets zusätzliche Lebenszyklusregeln hinzuzufügen.

In den folgenden Beispielen wird gezeigt, wie Sie Metriken zur Kostenoptimierung in Ihrem S3-Storage-Lens-Dashboard verwenden, um Ihre Speicherkosten zu optimieren.

Themen

Identifizieren Sie Ihre größten S3-Buckets

Sie zahlen für das Speichern von Objekten in S3-Buckets. Der Ihnen in Rechnung gestellte Preis hängt von der Größe Ihrer Objekte, der Speicherdauer der Objekte und ihren Speicherklassen ab. Mit S3 Storage Lens erhalten Sie einen zentralen Überblick über alle Buckets in Ihrem Konto. Um alle Buckets in allen Konten Ihrer Organisation anzuzeigen, können Sie ein AWS Organizations-Grad S3-Storage-Lens-Dashboard konfigurieren. In dieser Dashboard-Ansicht können Sie Ihre größten Buckets identifizieren.

Schritt 1: Identifizieren Ihrer größten Buckets

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.

   Wenn das Dashboard geöffnet wird, können Sie das Datum sehen, für das S3 Storage Lens zuletzt Metriken gesammelt hat. Ihr Dashboard wird immer für das letzte Datum geladen, für das Metriken verfügbar sind.

4. Scrollen Sie zum Abschnitt Top N overview for date (Top-N-Übersicht für Datum), um eine Rangfolge Ihrer größten Buckets nach der Metrik Total storage (Gesamtspeicher) für einen ausgewählten Zeitraum anzuzeigen.

   Sie können die Sortierreihenfolge ändern, um die kleinsten Buckets anzuzeigen. Sie können auch die Auswahl Metric (Metrik) anpassen, um Ihre Buckets nach einer der verfügbaren Metriken zu ordnen. Der Abschnitt Top N overview for date (Top-N-Übersicht für Datum) zeigt auch die prozentuale Veränderung gegenüber dem Vortag oder der Vorwoche sowie eine Sparkline zur Veranschaulichung des Trends an. Dieser Trend ist ein 14-Tage-Trend für kostenlose Metriken und ein 30-Tage-Trend für erweiterte Metriken und Empfehlungen.

   > **Note**
   >
   > Mit den erweiterten Metriken und Empfehlungen von S3 Storage Lens stehen Metriken 15 Monate für Abfragen zur Verfügung. Weitere Informationen finden Sie unter Metrikauswahl.

5. Für detailliertere Informationen zu Ihren Buckets scrollen Sie zum Seitenanfang und wählen Sie dann die Registerkarte Bucket.

   Auf der Registerkarte Buckets können Sie Details wie die aktuelle Wachstumsrate, die durchschnittliche Objektgröße, die größten Präfixe und die Anzahl der Objekte anzeigen.

Schritt 2: Navigieren zu Ihren Buckets und Untersuchen von Daten

Nachdem Sie Ihre größten S3-Buckets identifiziert haben, können Sie innerhalb der S3-Konsole zu jedem Bucket navigieren, um seine Objekte anzuzeigen, die damit verbundene Workload zu verstehen und interne Eigentümer des Buckets zu identifizieren. Sie können die Bucket-Eigentümer kontaktieren, um herauszufinden, ob das Wachstum erwartet wird oder ob es einer weiteren Überwachung und Kontrolle bedarf.

Entdecken Sie kalte Amazon S3-Buckets

Wenn Sie die fortschrittlichen Metriken von S3 Storage Lens aktiviert haben, können Sie Aktivitätsmetriken verwenden, um zu verstehen, wie kalt Ihre S3-Buckets sind. Ein „kalter“ Bucket ist ein Bucket, auf dessen Speicher nicht mehr (oder sehr selten) zugegriffen wird. Dieser Mangel an Aktivität weist normalerweise darauf hin, dass auf die Objekte des Buckets nicht häufig zugegriffen wird.

Aktivitätsmetriken wie GET-Anforderungen und Download-Bytes geben an, wie oft täglich auf Ihre Buckets zugegriffen wird. Um die Konsistenz des Zugriffsmusters zu verstehen und Buckets zu erkennen, auf die überhaupt nicht mehr zugegriffen wird, können Sie diese Daten über mehrere Monate hinweg trenden. Die Metrik der Abrufrate, die als Download-Bytes/Gesamtspeicher berechnet wird, gibt den Anteil des Speichers in einem Bucket an, auf den täglich zugegriffen wird.

> **Note**
>
> Download-Bytes werden dupliziert, wenn das gleiche Objekt mehrmals am Tag heruntergeladen wird.

Voraussetzung

Wenn Sie Aktivitätsmetriken in Ihrem S3-Storage-Lens-Dashboard anzeigen möchten, müssen Sie die Option Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) von S3 Storage Lens aktivieren und dann Activity metrics (Aktivitätsmetriken) auswählen. Weitere Informationen finden Sie unter Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards.

Schritt 1: Identifizieren aktiver Buckets

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.
4. Wählen Sie die Registerkarte Bucket aus und scrollen Sie dann nach unten zum Abschnitt Bubble analysis by buckets for date (Blasenanalyse nach Buckets für Datum).

   Im Abschnitt Bubble analysis by buckets for date (Blasenanalyse nach Buckets für Datum) können Sie Ihre Buckets in mehreren Dimensionen darstellen, indem Sie drei beliebige Metriken verwenden, um die X-axis (x-Achse), Y-axis (y-Achse) und die Size (Größe) der Blase darzustellen.
5. Wählen Sie für X-axis (x-Achse) Y-axis (y-Achse) und Size (Größe) die Metriken Total storage (Gesamtspeicher), % retrieval rate (% Abrufrate) und Average object size (Durchschnittliche Objektgröße) aus, um „kalte“ Buckets zu finden.

6. Suchen Sie im Abschnitt Bubble analysis by buckets for date (Blasenanalyse nach Buckets für Datum) nach Buckets mit Abrufraten von null (oder nahe null) und einer größeren relativen Speichergröße. Wählen Sie dann die Blase aus, die den Bucket darstellt.

   Es erscheint ein Feld mit Auswahlmöglichkeiten für detailliertere Einblicke. Führen Sie eine der folgenden Aktionen aus:

   a. Damit die Registerkarte Bucket aktualisiert wird und nur Metriken für den ausgewählten Bucket anzeigt, wählen Sie Drill down (Aufgliedern) und dann Apply (Anwenden) aus.
   b. Wenn Sie Ihre Daten auf Bucket-Ebene nach Konto, AWS-Region, Speicherklasse oder Bucket aggregieren möchten, wählen Sie Analyze by (Analysieren nach) aus und treffen Sie dann eine Auswahl für Dimension. Wenn Sie beispielsweise nach Speicherklasse aggregieren möchten, wählen Sie Storage class (Speicherklasse) für Dimension aus.

   Um Buckets zu finden, die kalt geworden sind, führen Sie eine Blasenanalyse mit den Metriken Gesamtspeicher, Abrufrate in % und Durchschnittliche Objektgröße durch. Suchen Sie nach Buckets mit Abrufraten von null (oder nahe null) und einer größeren relativen Speichergröße.

   Die Registerkarte Bucket Ihres Dashboards wird aktualisiert und zeigt Daten für die von Ihnen ausgewählte Aggregation oder den ausgewählten Filter an. Wenn Sie nach Speicherklasse oder einer anderen Dimension aggregiert haben, wird diese neue Registerkarte in Ihrem Dashboard geöffnet (z. B. die Registerkarte Storage class (Speicherklasse)).

Schritt 2: Untersuchen kalter Buckets

Von hier aus können Sie die Eigentümer von kalten Buckets in Ihrem Konto oder Ihrer Organisation identifizieren und herausfinden, ob dieser Speicher noch benötigt wird. Anschließend können Sie die Kosten optimieren, indem Sie Ablaufkonfigurationen für den Lebenszyklus für diese Buckets konfigurieren oder die Daten in einer der Amazon S3 Glacier-Speicherklassen archivieren.

Um das Problem von kalten Buckets in Zukunft zu vermeiden, können Sie Ihre Daten mithilfe von S3-Lifecycle-Konfigurationen für Ihre Buckets automatisch umstellen oder die automatische Archivierung mit S3 Intelligent-Tiering aktivieren.

Sie können Schritt 1 auch verwenden, um „heiße“ Buckets zu identifizieren. Anschließend können Sie sich vergewissern, dass diese Buckets die richtige S3-Speicherklasse verwenden, um sicherzustellen, dass sie ihre Anforderungen in Bezug auf Leistung und Kosten am effektivsten bearbeiten.

Suchen von unvollständigen mehrteiligen Uploads

Sie können mehrteilige Uploads verwenden, um sehr große Objekte (bis zu 5 TB) als einen Satz von Teilen hochzuladen und so den Durchsatz zu verbessern und Netzwerkprobleme schneller zu beheben. In Fällen, in denen der mehrteilige Upload-Prozess nicht abgeschlossen wird, verbleiben die unvollständigen Teile im Bucket (in einem unbrauchbaren Zustand). Für diese unvollständigen Teile fallen Speicherkosten an, bis der Upload-Prozess abgeschlossen ist oder die unvollständigen Teile entfernt werden. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Mit S3 Storage Lens können Sie die Anzahl der unvollständigen mehrteiligen Upload-Bytes in Ihrem Konto oder in Ihrem gesamten Unternehmen ermitteln. Dies umfasst auch mehrteilige Uploads, die älter sind als 7 Tage. Eine vollständige Liste der Metriken für unvollständige mehrteilige Uploads finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Als bewährte Methode empfehlen wir, Lebenszyklusregeln so zu konfigurieren, dass unvollständige mehrteilige Uploads, die älter als eine bestimmte Anzahl von Tagen sind, ablaufen. Wenn Sie Ihre Lebenszyklusregel so erstellen, dass unvollständige mehrteilige Uploads ablaufen, empfehlen wir 7 Tage als geeigneten Ausgangspunkt.

Schritt 1: Überprüfen allgemeiner Trends für unvollständige mehrteilige Uploads

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.
4. Wählen Sie im Abschnitt Snapshot for date (Snapshot für Datum) unter Metrics categories (Metrikkategorien) die Option Cost optimization (Kostenoptimierung) aus.

   Der Abschnitt Snapshot for date (Snapshot für Datum) wird aktualisiert und zeigt nun Metriken zu Cost optimization (Kostenoptimierung) an. Dazu gehören Incomplete multipart upload bytes greater than 7 days old (Unvollständige mehrteilige Upload-Bytes, die älter als 7 Tage sind).

   In jedem Diagramm Ihres S3-Storage-Lens-Dashboards können Sie Metriken für unvollständige mehrteilige Uploads sehen. Sie können diese Metriken verwenden, um die Auswirkungen unvollständiger mehrteiliger Upload-Bytes auf Ihren Speicher weiter zu bewerten, einschließlich deren Beitrag zu den allgemeinen Wachstumstrends. Sie können auch eine detaillierte Aufschlüsselung tieferer Aggregationsebenen aufrufen. Verwenden Sie dazu die Registerkarten

Account (Konto), AWS-Region, Bucket oder Storage class (Speicherklasse), um Ihre Daten eingehender zu analysieren. Ein Beispiel finden Sie unter Entdecken Sie kalte Amazon S3-Buckets.

Schritt 2: Identifizieren von Buckets, die die unvollständigsten mehrteiligen Upload-Bytes, aber keine Lebenszyklusregeln zum Abbruch von unvollständigen mehrteiligen Uploads haben

Voraussetzung

Wenn Sie in Ihrem S3-Storage-Lens-Dashboard die Metrik Abort incomplete multipart upload lifecycle rule count (Anzahl der Lebenszyklusregeln zum Abbrechen unvollständiger mehrteiliger Uploads) sehen möchten, müssen Sie Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) von S3 Storage Lens aktivieren und dann Advanced cost optimization metrics (Erweiterte Kostenoptimierungsmetriken) auswählen. Weitere Informationen finden Sie unter Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.
4. Wenn Sie bestimmte Buckets identifizieren möchten, in denen sich unvollständige mehrteilige Uploads ansammeln, die älter als 7 Tage sind, gehen Sie zum Abschnitt Top N overview for date (Top-N-Übersicht für Datum).

   Standardmäßig werden im Abschnitt Top N overview for date (Top-N-Übersicht für Datum) Metriken für die drei wichtigsten Buckets angezeigt. Sie können die Anzahl der Buckets im Feld Top N erhöhen oder verringern. Der Abschnitt Top N overview for date (Top-N-Übersicht für Datum) zeigt auch die prozentuale Veränderung gegenüber dem Vortag oder der Vorwoche sowie eine Sparkline zur Veranschaulichung des Trends an. (Dieser Trend ist ein 14-Tage-Trend für kostenlose Metriken und ein 30-Tage-Trend für erweiterte Metriken und Empfehlungen.)

   > **Note**
   >
   > Mit den erweiterten Metriken und Empfehlungen von S3 Storage Lens stehen Metriken 15 Monate für Abfragen zur Verfügung. Weitere Informationen finden Sie unter Metrikauswahl.

5. Wählen Sie für Metric (Metrik) in der Kategorie Cost optimization (Kostenoptimierung) die Option Incomplete multipart upload bytes greater than 7 days old (Unvollständige mehrteilige Upload-Bytes, die älter als 7 Tage sind) aus.

   Unter Top number buckets (Top N-Buckets) sehen Sie die Buckets mit den unvollständigsten mehrteiligen Upload-Speicherbytes, die älter als 7 Tage sind.

6. Wenn Sie detailliertere Metriken auf Bucket-Ebene für unvollständige mehrteilige Uploads anzeigen möchten, scrollen Sie zum Seitenanfang und wählen Sie dann die Registerkarte Bucket aus.

7. Scrollen Sie nach unten zum Abschnitt Buckets. Wählen Sie für Metrics categories (Metrikkategorien) die Option Cost optimization (Kostenoptimierung) aus. Löschen Sie dann Summary (Zusammenfassung).

   Die Liste Buckets wird aktualisiert und enthält alle verfügbaren Metriken für Cost optimization (Kostenoptimierung) für die angezeigten Buckets.

8. Wenn Sie die Liste Buckets so filtern möchten, dass nur bestimmte Kostenoptimierungsmetriken angezeigt werden, wählen Sie das Präferenzensymbol () aus.

9. Deaktivieren Sie die Schalter für alle Metriken zur Kostenoptimierung, bis nur noch Incomplete multipart upload bytes greater than 7 days old (Unvollständige mehrteilige Upload-Bytes, die älter als 7 Tage sind) und Abort incomplete multipart upload lifecycle rule count (Anzahl der Lebenszyklusregeln zum Abbrechen unvollständiger mehrteiliger Uploads) ausgewählt sind.

10. (Optional) Wählen Sie unter Page size (Seitengröße) die Anzahl der Buckets aus, die in der Liste angezeigt werden sollen.

11. Wählen Sie Bestätigen aus.

    Die Liste Buckets wird aktualisiert und zeigt Metriken auf Bucket-Ebene für unvollständige mehrteilige Uploads und die Anzahl der Lebenszyklusregeln an. Sie können diese Daten verwenden, um Buckets zu identifizieren, die die unvollständigsten mehrteiligen Upload-Bytes enthalten, die älter als 7 Tage sind und denen Lebenszyklusregeln fehlen, um unvollständige mehrteilige Uploads abzubrechen. Anschließend können Sie in der S3-Konsole zu diesen Buckets navigieren und Lebenszyklusregeln hinzufügen, um abgebrochene, unvollständige mehrteilige Uploads zu löschen.

Schritt 3: Hinzufügen einer Lebenszyklusregel, um unvollständige mehrteilige Uploads nach 7 Tagen zu löschen

Um unvollständige mehrteilige Uploads automatisch zu verwalten, können Sie die S3-Konsole verwenden, um eine Lebenszykluskonfiguration zu erstellen und unvollständige mehrteilige Upload-Bytes aus einem Bucket nach einer bestimmten Anzahl von Tagen ablaufen zu lassen. Weitere Informationen finden Sie unter Konfigurieren einer Bucket-Lebenszykluskonfiguration zum Löschen unvollständiger mehrteiliger Uploads.

Reduzieren Sie die Anzahl der beibehaltenen nicht aktuellen Versionen

Wenn die S3 Versioning aktiviert ist, werden mehrere Versionen desselben Objekts beibehalten, die zur schnellen Wiederherstellung von Daten verwendet werden können, wenn ein Objekt versehentlich gelöscht oder überschrieben wird. Wenn Sie S3 Versioning aktiviert haben, ohne Lebenszyklusregeln für die Umstellung oder den Ablauf nicht aktueller Versionen zu konfigurieren, kann sich eine große Anzahl nicht aktueller Versionen ansammeln, was sich auf die Speicherkosten auswirken kann. Weitere Informationen finden Sie unter Verwenden der Versioning in S3-Buckets.

Schritt 1: Identifizieren von Buckets mit den meisten nicht aktuellen Objektversionen

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.
4. Wählen Sie im Abschnitt Snapshot for date (Snapshot für Datum) unter Metrics categories (Metrikkategorien) die Option Cost optimization (Kostenoptimierung) aus.

   Der Abschnitt Snapshot for date (Snapshot für Datum) wird aktualisiert und zeigt nun Metriken für Cost optimization (Kostenoptimierung) an, zu denen auch die Metrik % noncurrent version bytes (% Bytes der nicht aktuellen Version) gehört. Die Metrik % noncurrent version bytes (% Bytes der nicht aktuellen Version) stellt den Anteil Ihrer gesamten Speicherbytes dar, der im Rahmen des Dashboards und für das ausgewählte Datum auf nicht aktuelle Versionen zurückzuführen ist.

   > **Note**
   >
   > Wenn der Wert % noncurrent version bytes (% Bytes der nicht aktuellen Version) mehr als 10 % Ihres Speicherplatzes auf Kontoebene beträgt, könnte dies ein Hinweis darauf sein, dass Sie zu viele Versionen speichern.

5. So identifizieren Sie bestimmte Buckets, in denen sich eine große Anzahl nicht aktueller Versionen ansammelt:

   a. Scrollen Sie nach unten zum Abschnitt Top N overview for date (Top-N-Übersicht für Datum). Geben Sie für Top N die Anzahl der Buckets ein, deren Daten Sie sehen möchten.

   b. Wählen Sie für Metric (Metrik) die Option % noncurrent version bytes (% Bytes der nicht aktuellen Version) aus.

      Unter Top number buckets (Top N-Buckets) werden die Buckets (für die von Ihnen angegebene Anzahl) mit dem höchsten Wert für % noncurrent version bytes (% Bytes der nicht aktuellen Version) angezeigt. Der Abschnitt Top N overview for date (Top-N-Übersicht für Datum) zeigt auch die prozentuale Veränderung gegenüber dem Vortag oder der Vorwoche sowie eine Sparkline zur Veranschaulichung des Trends an. Dieser Trend ist ein 14-Tage-Trend für kostenlose Metriken und ein 30-Tage-Trend für erweiterte Metriken und Empfehlungen.

      > **Note**
      >
      > Mit den erweiterten Metriken und Empfehlungen von S3 Storage Lens stehen Metriken 15 Monate für Abfragen zur Verfügung. Weitere Informationen finden Sie unter Metrikauswahl.

   c. Wenn Sie detailliertere Metriken auf Bucket-Ebene für nicht aktuelle Objektversionen anzeigen möchten, scrollen Sie zum Seitenanfang und wählen Sie dann die Registerkarte Bucket aus.

      In jedem Diagramm oder jeder Visualisierung Ihres S3-Storage-Lens-Dashboards können Sie mithilfe der Registerkarten Account (Konto), AWS-Region, Storage classe (Speicherklasse) oder Bucket tiefere Aggregationsebenen aufrufen. Ein Beispiel finden Sie unter Entdecken Sie kalte Amazon S3-Buckets.

   d. Wählen Sie im Abschnitt Buckets für Metric categories (Metrikkategorien) die Option Cost optimization (Kostenoptimierung) aus. Löschen Sie dann Summary (Zusammenfassung).

      Sie können jetzt die Metrik % noncurrent version bytes (% Bytes der nicht aktuellen Version) zusammen mit anderen Metriken sehen, die sich auf nicht aktuelle Versionen beziehen.

Schritt 2: Identifizieren von Buckets, für die Lebenszyklusregeln für Übergang und Ablauf nicht aktueller Versionen fehlen

Voraussetzung

Wenn Sie in Ihrem S3-Storage-Lens-Dashboard die Metriken Noncurrent version transition lifecycle rule count (Anzahl der Lebenszyklusregeln für den Übergang nicht aktueller Versionen) und Noncurrent version expiration lifecycle rule count (Anzahl der Lebenszyklusregeln für den Ablauf nicht aktueller Versionen) zu sehen, müssen Sie Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) von S3 Storage Lens aktivieren und dann Advanced cost optimization metrics (Erweiterte Kostenoptimierungsmetriken) auswählen. Weitere Informationen finden Sie unter Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.
4. Wählen Sie in Ihrem Storage-Lens-Dashboard die Registerkarte Bucket aus.
5. Scrollen Sie nach unten zum Abschnitt Buckets. Wählen Sie für Metrics categories (Metrikkategorien) die Option Cost optimization (Kostenoptimierung) aus. Löschen Sie dann Summary (Zusammenfassung).

   Die Liste Buckets wird aktualisiert und enthält alle verfügbaren Metriken für Cost optimization (Kostenoptimierung) für die angezeigten Buckets.
6. Wenn Sie die Liste Buckets so filtern möchten, dass nur bestimmte Kostenoptimierungsmetriken angezeigt werden, wählen Sie das Präferenzensymbol () aus.
7. Deaktivieren Sie die Schalter für alle Metriken zur Kostenoptimierung, bis nur noch die folgenden Metriken ausgewählt sind:

   - % noncurrent version bytes (% Bytes der nicht aktuellen Version)
   - Noncurrent version transition lifecycle rule count (Anzahl der Lebenszyklusregeln für den Übergang nicht aktueller Versionen)
   - Noncurrent version expiration lifecycle rule count (Anzahl der Lebenszyklusregeln für den Ablauf nicht aktueller Versionen)

8. (Optional) Wählen Sie unter Page size (Seitengröße) die Anzahl der Buckets aus, die in der Liste angezeigt werden sollen.
9. Wählen Sie Bestätigen aus.

   Die Liste Buckets wird aktualisiert und zeigt Metriken für die Anzahl der Bytes nicht aktueller Versionen und die Anzahl der Lebenszyklusregeln für nicht aktuelle Versionen an. Sie können diese Daten verwenden, um Buckets zu identifizieren, die einen hohen Prozentsatz an Bytes nicht aktueller Versionen enthalten, für die jedoch Übergangs- und Ablauflebenszyklusregeln fehlen. Anschließend können Sie in der S3-Konsole zu diesen Buckets navigieren und ihnen Lebenszyklusregeln hinzufügen.

Schritt 3: Hinzufügen von Lebenszyklusregeln für den Übergang oder den Ablauf nicht aktueller Objektversionen

Nachdem Sie festgestellt haben, welche Buckets eine weitere Untersuchung erfordern, können Sie in der S3-Konsole zu den Buckets navigieren und eine Lebenszyklusregel hinzufügen, um nicht aktuelle Versionen nach einer bestimmten Anzahl von Tagen ablaufen zu lassen. Um die Kosten zu senken und gleichzeitig nicht aktuelle Versionen beizubehalten, können Sie alternativ eine Lebenszyklusregel konfigurieren, um nicht aktuelle Versionen auf eine der Amazon S3 Glacier-Speicherklassen zu übertragen. Weitere Informationen finden Sie unter Beispiel 6: Spezifikation einer Lebenszyklus-Konfigurationsregel für einen Bucket mit Versioning .

Identifizieren von Buckets ohne Lebenszyklusregeln und Überprüfen der Anzahl von Lebenszyklusregeln

S3 Storage Lens bietet Metriken zur Anzahl der S3-Lebenszyklusregeln, anhand derer Sie Buckets identifizieren können, denen Lebenszyklusregeln fehlen. Um Buckets zu finden, die keine Lebenszyklusregeln haben, können Sie die Metrik Total buckets without lifecycle rules (Buckets insgesamt ohne Lebenszyklusregeln) verwenden. Ein Bucket ohne S3-Lifecycle-Konfiguration verfügt möglicherweise über Speicher, den Sie nicht mehr benötigen oder den Sie zu einer kostengünstigeren Speicherklasse migrieren können. Sie können auch Metriken zur Anzahl von Lebenszyklusregeln verwenden, um Buckets zu identifizieren, in denen bestimmte Arten von Lebenszyklusregeln fehlen, z. B. Ablauf- oder Übergangsregeln.

Voraussetzung

Wenn Sie in Ihrem S3-Storage-Lens-Dashboard die Metrik Total buckets without lifecycle rules (Buckets insgesamt ohne Lebenszyklusregeln) sehen möchten, müssen Sie Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) von S3 Storage Lens aktivieren und

dann Advanced cost optimization metrics (Erweiterte Kostenoptimierungsmetriken) auswählen. Weitere Informationen finden Sie unter Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards.

Schritt 1: Identifizieren von Buckets ohne Lebenszyklusregeln

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.
4. Wenn Sie bestimmte Buckets ohne Lebenszyklusregeln identifizieren möchten, scrollen Sie nach unten zum Abschnitt Top N overview for date (Top-N-Übersicht für Datum).

   Standardmäßig werden im Abschnitt Top N overview for date (Top-N-Übersicht für Datum) Metriken für die drei wichtigsten Buckets angezeigt. Im Feld Top N können Sie die Anzahl der Buckets erhöhen. Der Abschnitt Top N overview for date (Top-N-Übersicht für Datum) zeigt auch die prozentuale Veränderung gegenüber dem Vortag oder der Vorwoche sowie eine Sparkline zur Veranschaulichung des Trends an. Dieser Trend ist ein 14-Tage-Trend für kostenlose Metriken und ein 30-Tage-Trend für erweiterte Metriken und Empfehlungen.

   > **Note**
   >
   > Mit den erweiterten Metriken und Empfehlungen von S3 Storage Lens stehen Metriken 15 Monate für Abfragen zur Verfügung. Weitere Informationen finden Sie unter Metrikauswahl.

5. Wählen Sie für Metric (Metrik) in der Kategorie Cost optimization (Kostenoptimierung) die Option Total buckets without lifecycle rules (Buckets insgesamt ohne Lebenszyklusregeln) aus.
6. Überprüfen Sie die folgenden Daten für Total buckets without lifecycle rules (Buckets insgesamt ohne Lebenszyklusregeln):

   - Top number accounts (Top N-Konten) – Sehen Sie sich an, welche Konten die meisten Buckets ohne Lebenszyklusregeln haben.
   - Top number Regions (Top N-Regionen) – Sehen Sie sich eine Aufschlüsselung der Buckets ohne Lebenszyklusregeln nach Region an.
   - Top number buckets (Top N-Buckets) – Finden Sie heraus, für welche Buckets keine Lebenszyklusregeln festgelegt sind.

In jedem Diagramm oder jeder Visualisierung Ihres S3-Storage-Lens-Dashboards können Sie mithilfe der Registerkarten Account (Konto), AWS-Region, Storage classe (Speicherklasse) oder Bucket tiefere Aggregationsebenen aufrufen. Ein Beispiel finden Sie unter Entdecken Sie kalte Amazon S3-Buckets.

Nachdem Sie ermittelt haben, für welche Buckets keine Lebenszyklusregeln festgelegt sind, können Sie auch die Anzahl der spezifischen Lebenszyklusregeln für Ihre Buckets überprüfen.

Schritt 2: Überprüfen der Anzahl der Lebenszyklusregeln für Ihre Buckets

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie anzeigen möchten.
4. Wählen Sie in Ihrem S3-Storage-Lens-Dashboard die Registerkarte Bucket aus.
5. Scrollen Sie nach unten zum Abschnitt Buckets. Wählen Sie unter Metrics categories (Metrikkategorien) die Option Cost optimization (Kostenoptimierung) aus. Löschen Sie dann Summary (Zusammenfassung).

   Die Liste Buckets wird aktualisiert und enthält alle verfügbaren Metriken für Cost optimization (Kostenoptimierung) für die angezeigten Buckets.
6. Wenn Sie die Liste Buckets so filtern möchten, dass nur bestimmte Kostenoptimierungsmetriken angezeigt werden, wählen Sie das Präferenzensymbol () aus.
7. Deaktivieren Sie die Schalter für alle Metriken zur Kostenoptimierung, bis nur noch die folgenden Metriken ausgewählt sind:

   - Transition lifecycle rule count (Anzahl der Lebenszyklusregeln für den Übergang)
   - Expiration lifecycle rule count (Anzahl der Lebenszyklusregeln für den Ablauf)
   - Noncurrent version transition lifecycle rule count (Anzahl der Lebenszyklusregeln für den Übergang nicht aktueller Versionen)
   - Noncurrent version expiration lifecycle rule count (Anzahl der Lebenszyklusregeln für den Ablauf nicht aktueller Versionen)

- Abort incomplete multipart upload lifecycle rule count (Anzahl der Lebenszyklusregeln zum Abbrechen unvollständiger mehrteiliger Uploads)
- Total lifecycle rule count (Anzahl der gesamten Lebenszyklusregeln)

8. (Optional) Wählen Sie unter Page size (Seitengröße) die Anzahl der Buckets aus, die in der Liste angezeigt werden sollen.
9. Wählen Sie Bestätigen aus.

   Die Liste Buckets wird aktualisiert und zeigt die die Metriken für die Anzahl der Lebenszyklusregeln für Ihre Buckets an. Sie können diese Daten verwenden, um Buckets ohne Lebenszyklusregeln oder Buckets zu identifizieren, für die bestimmte Arten von Lebenszyklusregeln fehlen, z. B. Ablaufregeln oder Übergangsregeln. Anschließend können Sie in der S3-Konsole zu diesen Buckets navigieren und ihnen Lebenszyklusregeln hinzufügen.

Schritt 3: Hinzufügen von Lebenszyklusregeln

Nachdem Sie Buckets ohne Lebenszyklusregeln identifiziert haben, können Sie Lebenszyklusregeln hinzufügen. Weitere Informationen finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket und Beispiele der S3-Lebenszyklus-Konfiguration.

Verwenden von S3 Storage Lens zum Schutz Ihrer Daten

Sie können die Datenschutzmetriken von Amazon S3 Storage Lens verwenden, um Buckets zu identifizieren, in denen die bewährten Datenschutzmethoden nicht angewendet wurden. Sie können diese Metriken verwenden, um Maßnahmen zu ergreifen und Standardeinstellungen anzuwenden, die den bewährten Methoden zum Schutz Ihrer Daten in allen Buckets Ihres Kontos oder Ihrer Organisation entsprechen. Sie können beispielsweise Datenschutzmetriken verwenden, um Buckets, die keine AWS Key Management Service (AWS KMS)-Schlüssel (SSE-KMS) für die Standardverschlüsselung verwenden, oder Anforderungen, die AWS Signature Version 2 (SigV2) verwenden, zu identifizieren.

Die folgenden Anwendungsfälle bieten Strategien zur Verwendung Ihres S3-Storage-Lens-Dashboards, um Ausreißer zu ermitteln und bewährte Datenschutzmethoden auf Ihre S3-Buckets anzuwenden.

Themen

- Identifizieren von Buckets, die keine serverseitige Verschlüsselung mit AWS KMS für die Standardverschlüsselung (SSE-KMS) verwenden

- Identifizieren von Buckets, für die S3 Versioning aktiviert ist
- Identifizieren von Anforderungen, die AWS Signature Version 2 (SigV2) verwenden
- Zählen der Gesamtzahl der Replikationsregeln für jeden Bucket
- Identifizieren des Prozentsatz der Bytes mit Objektsperre

Identifizieren von Buckets, die keine serverseitige Verschlüsselung mit AWS KMS für die Standardverschlüsselung (SSE-KMS) verwenden

Mit der Amazon-S3-Standardverschlüsselung können Sie das Verhalten der Standardverschlüsselung für einen S3-Bucket festlegen. Weitere Informationen finden Sie unter the section called “Festlegen der Standard-Bucket-Verschlüsselung”.

Sie können die Metriken SSE-KMS enabled bucket count (SSE-KMS-fähige Bucket-Anzahl) und % SSE-KMS enabled buckets (% SSE-KMS-fähige Buckets) verwenden, um Buckets zu identifizieren, die serverseitige Verschlüsselung mit AWS KMS-Schlüsseln (SSE-KMS) als Standardverschlüsselung verwenden. S3 Storage Lens bietet auch Metriken für unverschlüsselte Bytes, unverschlüsselte Objekte, verschlüsselte Bytes und verschlüsselte Objekte. Eine vollständige Liste der Metriken finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Sie können SSE-KMS-Verschlüsselungsmetriken im Kontext allgemeiner Verschlüsselungsmetriken analysieren, um Buckets zu identifizieren, die SSE-KMS nicht verwenden. Wenn Sie SSE-KMS für alle Buckets in Ihrem Konto oder Ihrer Organisation verwenden möchten, können Sie die Standardverschlüsselungseinstellungen für diese Buckets aktualisieren, um SSE-KMS zu verwenden. Zusätzlich zu SSE-KMS können Sie auch die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) oder vom Kunden bereitgestellten Schlüsseln (SSE-C) verwenden. Weitere Informationen finden Sie unter Datenschutz durch Verschlüsselung.

Schritt 1: Identifizieren, welche Buckets SSE-KMS für die Standardverschlüsselung verwenden

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. Wählen Sie im Abschnitt Trends and distributions (Trends und Verteilungen) die % SSE-KMS enabled bucket count (% der Anzahl der SSE-KMS-fähigen Buckets) als primäre Metrik und % encrypted bytes (% verschlüsselte Bytes) als sekundäre Metrik aus.

Das Diagramm Trend for date (Trend für Datum) wird aktualisiert und zeigt Trends für SSE-KMS und verschlüsselte Bytes an.

5. So zeigen Sie detailliertere Einblicke auf Bucket-Ebene für SSE-KMS an:

   a. Wählen Sie einen Punkt im Diagramm aus. Es erscheint ein Feld mit Auswahlmöglichkeiten für detailliertere Einblicke.

   b. Wählen Sie die Dimension Buckets aus. Wählen Sie dann Apply (Anwenden).

6. Wählen Sie im Diagramm Distribution by buckets for date (Verteilung nach Buckets für Datum) die Metrik SSE-KMS enabled bucket count (SSE-KMS-fähige Bucket-Anzahl) aus.

7. Sie können jetzt sehen, für welche Buckets SSE-KMS aktiviert ist und für welche nicht.

Schritt 2: Aktualisieren der Standardverschlüsselungseinstellungen für den Bucket

Nachdem Sie nun im Kontext Ihrer Metrik % encrypted bytes (% verschlüsselte Bytes) ermittelt haben, welche Buckets SSE-KMS verwenden, können Sie Buckets identifizieren, die SSE-KMS nicht verwenden. Sie können dann optional in der S3-Konsole zu diesen Buckets navigieren und deren Standardverschlüsselungseinstellungen aktualisieren, um SSE-KMS oder SSE-S3 zu verwenden. Weitere Informationen finden Sie unter Konfigurieren der Standardverschlüsselung.

Identifizieren von Buckets, für die S3 Versioning aktiviert ist

Wenn die S3-Versioningfunktion aktiviert ist, behält sie mehrere Versionen desselben Objekts bei, die zur schnellen Wiederherstellung von Daten verwendet werden können, wenn ein Objekt versehentlich gelöscht oder überschrieben wird. Sie können die Metrik Versioning-enabled bucket count (Bucket-Anzahl mit aktiviertem Versioning) verwenden, um zu sehen, welche Buckets S3 Versioning verwenden. Anschließend können Sie in der S3-Konsole Maßnahmen ergreifen, um S3 Versioning für andere Buckets zu aktivieren.

Schritt 1: Identifizieren von Buckets, für die S3 Versioning aktiviert ist

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Gehen Sie im Navigationsbereich auf Storage Lens und dann auf Dashboards.

3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.

4. Wählen Sie im Abschnitt Trends and distributions (Trends und Verteilungen) die Metrik Versioning-enabled bucket count (Bucket-Anzahl mit aktiviertem Versioning) als primäre Metrik und Buckets als sekundäre Metrik aus.

   Das Diagramm Trend for date (Trend für Datum) wird aktualisiert, um Trends für Buckets mit aktiviertem S3 Versioning anzuzeigen. Direkt unter der Trendlinie sehen Sie die Unterabschnitte Storage class distribution (Speicherklassenverteilung) und Region distribution (Regionale Verteilung).

5. Gehen Sie wie folgt vor, um detailliertere Einblicke für einen der Buckets im Diagramm Trend for date (Trend für Datum) zu sehen und eine eingehendere Analyse durchführen zu können:

   a. Wählen Sie einen Punkt im Diagramm aus. Es erscheint ein Feld mit Auswahlmöglichkeiten für detailliertere Einblicke.

   b. Wählen Sie eine Dimension aus, die Sie für eine eingehendere Analyse auf Ihre Daten anwenden möchten: Account (Konto), AWS-Region, Storage class (Speicherklasse) oder Bucket. Wählen Sie dann Apply (Anwenden).

6. Wählen Sie im Bubble analysis by buckets for date (Blasenanalyse nach Buckets für Datum) die Metriken Versioning-enabled bucket count (Bucket-Anzahl mit aktiviertem Versioning), Buckets und Active Buckets (Aktive Buckets) aus.

   Der Abschnitt Bubble analysis by buckets for date (Blasenanalyse nach Buckets für Datum) wird aktualisiert und zeigt nun Daten für die ausgewählten Metriken an. Sie können diese Daten verwenden, um zu sehen, für welche Buckets S3 Versioning im Kontext Ihrer gesamten Bucket-Anzahl aktiviert ist. Im Abschnitt Bubble analysis by buckets for date (Blasenanalyse nach Buckets für Datum) können Sie Ihre Buckets in mehreren Dimensionen darstellen, indem Sie drei beliebige Metriken verwenden, um die X-axis (x-Achse), Y-axis (y-Achse) und die Size (Größe) der Blase darzustellen.

Schritt 2: Aktivieren von S3 Versioning

Nachdem Sie Buckets identifiziert haben, für die S3 Versioning aktiviert ist, können Sie Buckets identifizieren, für die S3 Versioning noch nie aktiviert war oder deren Versionsverwaltung ausgesetzt ist. Anschließend können Sie die Versionsverwaltung für diese Buckets in der S3-Konsole optional aktivieren. Weitere Informationen finden Sie unter Aktivieren des Versioning für Buckets.

Identifizieren von Anforderungen, die AWS Signature Version 2 (SigV2) verwenden

Sie können die Metrik All unsupported signature requests (Alle nicht unterstützten Signaturanforderungen) verwenden, um Anfragen zu identifizieren, die AWS Signature Version 2 (SigV2) verwenden. Diese Daten können Ihnen helfen, bestimmte Anwendungen zu identifizieren, die SigV2 verwenden. Sie können diese Anwendungen dann zu AWS Signature Version 4 (SigV4) migrieren.

SigV4 ist die empfohlene Signaturmethode für alle neuen S3-Anwendungen. SigV4 bietet eine verbesserte Sicherheit und wird in allen AWS-Regionen unterstützt. Weitere Informationen finden Sie unter Amazon S3 Update – SigV2 Deprecation Period Extended and Modified.

Voraussetzung

Zum Anzeigen von All unsupported signature requests (Alle nicht unterstützten Signaturanforderungen) in Ihrem S3-Storage-Lens-Dashboard müssen Sie die Option Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) von S3 Storage Lens aktivieren und dann Advanced data protection metrics (Erweiterte Datensicherheitsmetriken) auswählen. Weitere Informationen finden Sie unter Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards.

Schritt 1: Untersuchen der SigV2-Signaturtrends nach AWS-Konto, Region und Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. So identifizieren Sie bestimmte Buckets, Konten und Regionen mit Anforderungen, die SigV2 verwenden:

    a. Geben Sie unter Top N overview for date (Top-N-Übersicht für Datum) im Feld Top N die Anzahl der Buckets ein, deren Daten Sie sehen möchten.

    b. Wählen Sie für Metric (Metrik) die Option All unsupported signature requests (Alle nicht unterstützten Signaturanforderungen) in der Kategorie Data protection (Datenschutz) aus.

    Die Ansicht Top N overview for date (Top-N-Übersicht für Datum) wird mit den Daten für SigV2-Anforderungen nach Konto, AWS-Region und Bucket aktualisiert. Der Abschnitt Top N overview for date (Top-N-Übersicht für Datum) zeigt auch die prozentuale Veränderung

gegenüber dem Vortag oder der Vorwoche sowie eine Sparkline zur Veranschaulichung des Trends an. Dieser Trend ist ein 14-Tage-Trend für kostenlose Metriken und ein 30-Tage-Trend für erweiterte Metriken und Empfehlungen.

> **Note**
>
> Mit den erweiterten Metriken und Empfehlungen von S3 Storage Lens stehen Metriken 15 Monate für Abfragen zur Verfügung. Weitere Informationen finden Sie unter Metrikauswahl.

Schritt 2: Identifizieren von Buckets, auf die Anwendungen über SigV2-Anforderungen zugreifen

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. Wählen Sie in Ihrem Storage-Lens-Dashboard die Registerkarte Bucket aus.
5. Scrollen Sie nach unten zum Abschnitt Buckets. Wählen Sie unter Metrics categories (Metrikkategorien) die Option Data protection (Datenschutz) aus. Löschen Sie dann Summary (Zusammenfassung).

   Die Liste Buckets wird aktualisiert und enthält alle verfügbaren Metriken für Data protection (Datenschutz) für die angezeigten Buckets.
6. Wenn Sie die Liste Buckets so filtern möchten, dass nur bestimmte Datenschutzmetriken angezeigt werden, wählen Sie das Präferenzensymbol () aus.
7. Deaktivieren Sie die Schalter für alle Datenschutzmetriken, bis nur noch die folgenden Metriken ausgewählt sind:

   - All unsupported signature requests (Alle nicht unterstützten Signaturanforderungen)
   - % all unsupported signature requests (% aller nicht unterstützten Signaturanforderungen)
8. (Optional) Wählen Sie unter Page size (Seitengröße) die Anzahl der Buckets aus, die in der Liste angezeigt werden sollen.

9. Wählen Sie Bestätigen aus.

   Die Liste Buckets wird aktualisiert, um Metriken auf Bucket-Ebene für SigV2-Anforderungen anzuzeigen. Sie können diese Daten verwenden, um bestimmte Buckets zu identifizieren, die SigV2-Anforderungen haben. Anschließend können Sie diese Informationen verwenden, um Ihre Anwendungen zu SigV4 zu migrieren. Weitere Informationen finden Sie unter Authenticating Requests (Authentifizierung von Anforderungn) (AWS Signature Version 4) in der API-Referenz für Amazon Simple Storage Service.

Zählen der Gesamtzahl der Replikationsregeln für jeden Bucket

S3 Replication ermöglicht das automatische, asynchrone Kopieren von Objekten in Amazon-S3-Buckets. Buckets, die für die Objektreplikation konfiguriert sind, können sich im Besitz desselben AWS-Konto oder unterschiedlicher Konten befinden. Weitere Informationen finden Sie unter Replizieren von Objekten.

Sie können die Metriken zur Anzahl der Replikationsregeln von S3 Storage Lens verwenden, um detaillierte Informationen pro Bucket zu Ihren Buckets zu erhalten, die für die Replikation konfiguriert sind. Zu diesen Informationen gehören Replikationsregeln innerhalb von Buckets und Regionen sowie Bucket- und regionsübergreifende Replikationsregeln.

Voraussetzung

Zum Anzeigen von Metriken zur Anzahl der Replikationsregeln in Ihrem S3-Storage-Lens-Dashboard müssen Sie die Option Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) von S3 Storage Lens aktivieren und dann Advanced data protection metrics (Erweiterte Datensicherheitsmetriken) auswählen. Weitere Informationen finden Sie unter Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards.

Schritt 1: Zählen der Gesamtzahl der Replikationsregeln für jeden Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. Wählen Sie in Ihrem Storage-Lens-Dashboard die Registerkarte Bucket aus.

5. Scrollen Sie nach unten zum Abschnitt Buckets. Wählen Sie unter Metrics categories (Metrikkategorien) die Option Data protection (Datenschutz) aus. Löschen Sie dann Summary (Zusammenfassung).
6. Wenn Sie die Liste Buckets so filtern möchten, dass nur Metriken zur Anzahl der Replikationsregeln angezeigt werden, wählen Sie das Präferenzensymbol () aus.
7. Deaktivieren Sie die Schalter für alle Datenschutzmetriken, bis nur noch die Metriken zur Anzahl der Replikationsregeln ausgewählt sind:
   - Same-Region Replication rule count (Anzahl der Replikationsregeln für dieselbe Region)
   - Cross-Region Replication rule count (Anzahl der regionsübergreifenden Replikationsregeln)
   - Same-account replication rule count (Anzahl der Replikationsregeln für dasselbe Konto)
   - Cross-account replication rule count (Kontenübergreifende Replikationsregelanzahl)
   - Total replication rule count (Gesamte Replikationsregelnanzahl)
8. (Optional) Wählen Sie unter Page size (Seitengröße) die Anzahl der Buckets aus, die in der Liste angezeigt werden sollen.
9. Wählen Sie Bestätigen aus.

Schritt 2: Hinzufügen von Replikationsregeln

Nachdem Sie die Anzahl der Replikationsregeln pro Bucket festgelegt haben, können Sie optional weitere Replikationsregeln erstellen. Weitere Informationen finden Sie unter Anleitungen: Beispiele zum Konfigurieren der Replikation.

Identifizieren des Prozentsatz der Bytes mit Objektsperre

Mit der S3-Objektsperre können Sie Objekte anhand des Modells write-once-read-many (WORM) speichern. Mit der Objektsperre können Sie verhindern, dass Objekte für einen bestimmten Zeitraum oder auf unbestimmte Zeit gelöscht oder überschrieben werden. Sie können die Objektsperre nur aktivieren, wenn Sie einen Bucket erstellen und S3 Versioning aktivieren. Sie können jedoch den Aufbewahrungszeitraum für einzelne Objektversionen bearbeiten oder gesetzliche Aufbewahrungsfristen für Buckets anwenden, für die die Objektsperre aktiviert ist. Weitere Informationen finden Sie unter Verwenden der S3-Objektsperre.

Sie können Metriken für die Objektsperre in S3 Storage Lens verwenden, um die Metrik % Object Lock bytes (% Bytes mit Objektsperre) für Ihr Konto oder Ihre Organisation anzuzeigen. Sie können diese Informationen verwenden, um Buckets in Ihrem Konto oder Ihrer Organisation zu identifizieren, die Ihre bewährten Datenschutzmethoden nicht einhalten.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. Wählen Sie im Abschnitt Snapshot unter Metrics categories (Metrikkategorien) die Option Data protection (Datenschutz) aus.

   Der Abschnitt Snapshot wird aktualisiert und zeigt nun Datenschutzmetriken an, einschließlich der Metrik % Object Lock bytes (% Bytes mit Objektsperre). Sie können den Gesamtprozentsatz der Bytes mit Objektsperre für Ihr Konto oder Ihre Organisation sehen.
5. Wenn Sie die Metrik % Object Lock bytes (% Bytes mit Objektsperre) pro Bucket sehen möchten, scrollen Sie nach unten zum Abschnitt Top N overview (Top-N-Übersicht).

   Wenn Sie Daten mit Objektsperre auf Objektebene abrufen möchten, können Sie auch die Metriken Object Lock object count (Objektsperren-Objektanzahl) und % Object Lock objects (% Objekte mit Objektsperre) verwenden.
6. Wählen Sie für Metric (Metrik) die Option % Object Lock Bytes (% Bytes mit Objektsperre) aus der Kategorie Data protection (Datenschutz) aus.

   Standardmäßig werden im Abschnitt Top N overview for date (Top-N-Übersicht für Datum) Metriken für die drei wichtigsten Buckets angezeigt. Im Feld Top N können Sie die Anzahl der Buckets erhöhen. Der Abschnitt Top N overview for date (Top-N-Übersicht für Datum) zeigt auch die prozentuale Veränderung gegenüber dem Vortag oder der Vorwoche sowie eine Sparkline zur Veranschaulichung des Trends an. Dieser Trend ist ein 14-Tage-Trend für kostenlose Metriken und ein 30-Tage-Trend für erweiterte Metriken und Empfehlungen.

> **Note**
>
> Mit den erweiterten Metriken und Empfehlungen von S3 Storage Lens stehen Metriken 15 Monate für Abfragen zur Verfügung. Weitere Informationen finden Sie unter Metrikauswahl.

7. Überprüfen Sie die folgenden Daten für % Object Lock bytes (% Objekte mit Objektsperre):

   - Top number accounts (Top N-Konten) – Sehen Sie sich an, welche Konten den höchsten und den niedrigsten Wert für % Object Lock bytes (% Bytes mit Objektsperre) haben.
   - Top number Regions (Top N-Regionen) – Zeigt eine Aufschlüsselung der % Object Lock bytes (% Bytes mit Objektsperre) nach Region an.
   - Top number buckets (Top N-Buckets) – Sehen Sie sich an, welche Buckets den höchsten und den niedrigsten Wert für % Object Lock bytes (% Bytes mit Objektsperre) haben.

Verwenden von S3 Storage Lens zur Überwachung der Einstellungen für die Objekteigentümerschaft

Amazon S3 Object Ownership ist eine S3-Einstellung auf Bucket-Ebene, mit der Sie Zugriffssteuerungslisten (ACLs) deaktivieren und die Eigentümerschaft der Objekte in Ihrem Bucket kontrollieren können. Wenn Sie die Einstellung für die Objekteigentümerschaft auf „Bucket owner enforced“ (Bucket-Eigentümer erzwungen) festlegen, können Sie Zugriffssteuerungslisten (ACLs) deaktivieren und die Eigentümerschaft für alle Objekte in Ihrem Bucket übernehmen. Dieser Ansatz vereinfacht die Zugriffsverwaltung für in Amazon S3 gespeicherte Daten.

Wenn ein anderes AWS-Konto ein Objekt in Ihren S3-Bucket hochlädt, besitzt dieses Konto (der Objektschreiber) standardmäßig das Objekt, hat Zugriff darauf und kann anderen Benutzern über ACLs Zugriff darauf gewähren. Sie können Object Ownership verwenden, um dieses Standardverhalten zu ändern.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen daher, ACLs zu deaktivieren, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn Sie die Objekteigentümerschaft auf „Bucket owner enforced“ (Bucket-Eigentümer erzwungen) festlegen, können Sie ACLs deaktivieren und sich auf Richtlinien für die Zugriffskontrolle verlassen. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

Mit den Zugriffsverwaltungsmetriken von S3 Storage Lens können Sie Buckets identifizieren, für die ACLs nicht deaktiviert sind. Nachdem Sie diese Buckets identifiziert haben, können Sie ACL-Berechtigungen zu Richtlinien migrieren und die ACLs für diese Buckets deaktivieren.

Themen

Schritt 1: Identifizieren allgemeiner Trends für die Einstellungen der Objekteigentümerschaft

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. Wählen Sie im Abschnitt Snapshot for date (Snapshot für Datum) unter Metrics categories (Metrikkategorien) die Option Access management (Zugriffsverwaltung) aus.

   Der Abschnitt Snapshot for date (Snapshot für Datum) wird aktualisiert und zeigt nun die Metrik % Object Ownership bucket owner enforced (% Bucket-Eigentümer erzwungen für Objekteigentümerschaft) an. Sie können den Gesamtanteil an Buckets in Ihrem Konto oder Ihrer Organisation anzeigen, die die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft verwenden, um ACLs zu deaktivieren.

Schritt 2: Identifizieren von Trends auf Bucket-Ebene für die Einstellungen der Objekteigentümerschaft

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.

4. Wenn Sie detailliertere Metriken auf Bucket-Ebene anzeigen möchten, wählen Sie die Registerkarte Bucket aus.
5. Wählen Sie im Abschnitt Distribution by buckets for date (Verteilung nach Buckets für Datum) die Metrik % Object Ownership bucket owner enforced (% Bucket-Eigentümer erzwungen für Objekteigentümerschaft) aus.

   Das Diagramm wird aktualisiert und zeigt nun eine Aufschlüsselung je Bucket für % Object Ownership bucket owner enforced (% Bucket-Eigentümer erzwungen für Objekteigentümerschaft) an. Sie können sehen, welche Buckets die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft verwenden, um ACLs zu deaktivieren.
6. Wenn Sie die Einstellungen „Bucket-Eigentümer erzwungen“ im Kontext sehen möchten, scrollen Sie nach unten zum Abschnitt Buckets. Wählen Sie für Metrics categories (Metrikkategorien) die Option Access management (Zugriffsverwaltung) aus. Löschen Sie dann Summary (Zusammenfassung).

   In der Liste Buckets werden Daten für alle drei Einstellungen für die Objekteigentümerschaft angezeigt: „Bucket owner enforced“ (Bucket-Eigentümer erzwungen), „Bucket owner preferred (Bucket-Eigentümer bevorzugt und „Object writer“ (Objektschreiber).
7. Wenn Sie die Liste Buckets so filtern möchten, dass Metriken nur für eine bestimmte Einstellung der Objekteigentümerschaft angezeigt werden, wählen Sie das Präferenzensymbol ().
8. Löschen Sie die Metriken, die Sie nicht anzeigen möchten.
9. (Optional) Wählen Sie unter Page size (Seitengröße) die Anzahl der Buckets aus, die in der Liste angezeigt werden sollen.
10. Wählen Sie Confirm (Bestätigen).

Schritt 3: Aktualisieren Ihrer Einstellung für die Objekteigentümerschaft auf „Bucket owner enforced“ (Bucket-Eigentümer erzwungen) zur Deaktivierung von ACLs

Nachdem Sie Buckets identifiziert haben, die die Einstellungen „Object writer“ (Objektschreiber) und „Bucket owner preferred“ (Bucket-Eigentümer bevorzugt) für die Objekteigentümerschaft verwenden, können Sie Ihre ACL-Berechtigungen zu Bucket-Richtlinien migrieren. Wenn Sie die Migration Ihrer ACL-Berechtigungen abgeschlossen haben, können Sie Ihre Einstellungen für die Objekteigentümerschaft auf „Bucket owner enforced“ (Bucket-Eigentümer erzwungen) aktualisieren, um ACLs zu deaktivieren. Weitere Informationen finden Sie unter Voraussetzungen für die Deaktivierung von ACLs.

Verwenden der Metriken von S3 Storage Lens zur Verbesserung der Leistung

Wenn Sie die Option S3 Storage Lens advanced metrics (Erweiterte Metriken von S3 Storage Lens) aktiviert haben, können Sie die detaillierten Statuscode-Metriken verwenden, um die Anzahl von erfolgreichen oder fehlgeschlagenen Anforderungen zu erhalten. Sie können diese Informationen verwenden, um Probleme mit dem Zugriff oder der Leistung zu beheben. Detaillierte Statuscodemetriken zeigen die Anzahl der HTTP-Statuscodes an, z. B. 403 Forbidden und 503 Service Unavailable. Sie können allgemeine Trends anhand detaillierter Statuscode-Metriken für S3-Buckets, Konten und Organisationen untersuchen. Anschließend können Sie sich Metriken auf Bucket-Ebene genauer ansehen, um Workloads zu identifizieren, die derzeit auf diese Buckets zugreifen und Fehler verursachen.

Sie können sich beispielsweise die Metrik 403 Forbidden error count (Anzahl der Fehler 403 Forbidden) ansehen, um Workloads zu identifizieren, die auf Buckets zugreifen, ohne dass die korrekten Berechtigungen angewendet wurden. Nachdem Sie diese Workloads identifiziert haben, können Sie außerhalb von S3 Storage Lens detaillierte Einblicke erhalten, um Ihre „403 Forbidden“-Fehler zu beheben.

Dieses Beispiel zeigt Ihnen, wie Sie eine Trendanalyse für den Fehler „403 Forbidden“ durchführen, indem Sie die Metriken 403 Forbidden error count (Anzahl der Fehler 403 Forbidden) und % 403 Forbidden errors (% Fehler 403 Forbidden) verwenden. Anhand dieser Metriken können Sie Workloads identifizieren, die auf Buckets zugreifen, ohne dass die korrekten Berechtigungen angewendet wurden. Eine ähnliche Trendanalyse können Sie für jede der anderen Detailed status code metrics (Detaillierte Statuscode-Metriken) durchführen. Weitere Informationen finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Voraussetzung

Wenn Sie Detailed status code metrics (Detaillierte Statuscode-Metriken) in Ihrem S3-Storage-Lens-Dashboard anzeigen möchten, müssen Sie die Option Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) von S3 Storage Lens aktivieren und dann Detailed status code metrics (Detaillierte Statuscode-Metriken) auswählen. Weitere Informationen finden Sie unter Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards.

Themen

Schritt 1: Durchführen einer Trendanalyse für einen einzelnen HTTP-Statuscode

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. Wählen Sie im Abschnitt Trends and distributions (Trends und Verteilungen) für Primary metric (Primäre Metrik) in der Kategorie Detailed status codes (Detaillierte Statuscodes) die Option 403 Forbidden error count (Anzahl der Fehler 403 Forbidden) aus. Wählen Sie für Secondary metric (Sekundäre Metrik) die Option % 403 Forbidden errors (% Fehler 403 Forbidden) aus.
5. Scrollen Sie nach unten zum Abschnitt Top N overview for date (Top-N-Übersicht für Datum). Wählen Sie für Metrics (Metriken) die Option 403 Forbidden error count (Anzahl der Fehler 403 Forbidden) oder % 403 Forbidden errors (% Fehler 403 Forbidden) in der Kategorie Detailed status codes (Detaillierte Statuscodes) aus.

   Der Abschnitt Top N overview for date (Top-N-Übersicht für Datum) wird aktualisiert und zeigt nun die Anzahl der häufigsten Fehler „403 Forbidden“ nach Konto, AWS-Region und Bucket an.

Schritt 2: Analysieren der Fehleranzahl nach Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards den Namen des Dashboards aus, das Sie anzeigen möchten.
4. Wählen Sie in Ihrem Storage-Lens-Dashboard die Registerkarte Bucket aus.
5. Scrollen Sie nach unten zum Abschnitt Buckets. Wählen Sie für Metrics categories (Metrikkategorien) die Option Detailed status code metrics (Detaillierte Statuscode-Metriken) aus. Löschen Sie dann Summary (Zusammenfassung).

   Die Liste Buckets wird aktualisiert und zeigt alle verfügbaren detaillierten Statuscode-Metriken an. Sie können diese Informationen verwenden, um zu sehen, welche Buckets einen großen Anteil bestimmter HTTP-Statuscodes enthalten und welche Statuscodes für die Buckets gängig sind.

6. Wenn Sie die Liste Buckets so filtern möchten, dass nur bestimmte detaillierte Statuscode-Metriken angezeigt werden, wählen Sie das Präferenzensymbol ().
7. Deaktivieren Sie die Schalter für alle detaillierten Statuscode-Metriken, die Sie nicht in der Liste Buckets anzeigen möchten.
8. (Optional) Wählen Sie unter Page size (Seitengröße) die Anzahl der Buckets aus, die in der Liste angezeigt werden sollen.
9. Wählen Sie Confirm (Bestätigen).

   In der Liste Buckets werden Metriken zur Fehleranzahl für die von Ihnen angegebene Anzahl von Buckets angezeigt. Sie können diese Informationen verwenden, um bestimmte Buckets zu identifizieren, in denen viele Fehler auftreten, und um Fehler nach Bucket zu beheben.

Schritt 3: Fehlerbehebung

Nachdem Sie Buckets mit einem hohen Anteil an bestimmten HTTP-Statuscodes identifiziert haben, können Sie diese Fehler beheben. Weitere Informationen finden Sie unter:

- Warum erhalte ich die Fehlermeldung „403 Forbidden“, wenn ich versuche, Dateien in Amazon S3 hochzuladen?
- Warum erhalte ich die Fehlermeldung „403 Forbidden“, wenn ich versuche, eine Bucket-Richtlinie in Amazon S3 zu ändern?
- Wie behebe ich „403 Forbidden“-Fehler in meinem Amazon-S3-Bucket, bei dem alle Ressourcen aus denselben AWS-Konto stammen?
- Wie behebe ich einen HTTP 500- oder 503-Fehler von Amazon S3?

Amazon S3-Storage-Lens-Metrikglossar

Das Metrikglossar von Amazon S3 Storage Lens enthält eine vollständige Liste der kostenlosen und erweiterten Metriken für S3 Storage Lens.

S3 Storage Lens bietet kostenlose Metriken für alle Dashboards und Konfigurationen mit der Option zum Upgrade auf erweiterte Metriken.

- Kostenlose Metriken enthalten relevante Metriken für Ihre Speichernutzung, z. B. die Anzahl der Buckets und der Objekte in Ihrem Konto. Zu den kostenlosen Metriken gehören auch

auf Anwendungsfällen basierende Metriken, etwa Metriken zur Kostenoptimierung und zum Datenschutz. Alle kostenlosen Metriken werden täglich erfasst und Daten stehen für Abfragen bis zu 14 Tage lang Daten zur Verfügung.

- Zu den erweiterten Metriken und Empfehlungen gehören neben den kostenlosen Metriken weitere Metriken, wie erweiterte Metriken zum Datenschutz und zur Kostenoptimierung. Zu den erweiterten Metriken gehören auch zusätzliche Metrikkategorien wie Aktivitätsmetriken und detaillierte Statuscode-Metriken. Erweiterte Metrikdaten stehen 15 Monate für Abfragen zur Verfügung.

  Für die Verwendung von S3 Storage Lens mit fortschrittlichen Metriken und Empfehlungen fallen zusätzliche Gebühren an. Weitere Informationen finden Sie unter Amazon S3 – Preise. Weitere Informationen über erweiterte Metriken und Empfehlungen finden Sie unter Metrikauswahl.

  > **Note**
  >
  > Für Storage-Lens-Gruppen sind nur Speichermetriken des kostenlosen Kontingents verfügbar. Metriken für die Advanced-Stufe sind auf Storage-Lens-Gruppenebene nicht verfügbar.

Metriknamen

Die Spalte Metrikname in der folgenden Tabelle enthält den Namen jeder S3 Storage Lens in der S3-Konsole. Die Spalte CloudWatch und Export enthält den Namen jeder Metrik in Amazon CloudWatch und die Metrikexportdatei, die Sie in Ihrem S3-Storage-Lens-Dashboard konfigurieren können.

Abgeleitete Metrikformeln

Abgeleitete Metriken sind für den Export von Metriken und die CloudWatch-Veröffentlichungsoption nicht verfügbar. Sie können jedoch die in der Spalte abgeleitete Metrikformeln angegebenen Metrikformeln verwenden, um sie zu berechnen.

Interpretieren der Präfix-Symbole von Amazon S3 Storage Lens für Vielfache von Metrikeinheiten (K, M, G usw.)

Metrikeinheiten von S3 Storage Lens werden mit mehreren Präfix-Symbolen geschrieben. Diese Präfixsymbole entsprechen den Symbolen des Internationalen Einheitensystems (SI), die vom Internationalen Büro für Maß und Gewicht (BIPM) standardisiert sind. Diese Symbole werden auch im Unified Code for Units of Measure (Einheitlicher Code für Maßeinheiten, UCUM) verwendet. Weitere Informationen finden Sie unter Liste der SI-Präfix-Symbole.

Note

- Die Maßeinheit für S3-Speicherbytes ist in binären Gigabyte (GB) angegeben, wobei 1 GB 2^{30} Byte, 1 TB 2^{40} Byte und 1 PB 2^{50} Byte entspricht. Diese Maßeinheit wird auch als Gibibyte (GiB) bezeichnet, entsprechend der Definition der International Electrotechnical Commission (IEC).
- Wenn ein Objekt basierend auf seiner Lebenszykluskonfiguration das Ende seiner Lebensdauer erreicht hat, stellt Amazon S3 das Objekt zum Entfernen in eine Warteschlange und entfernt es asynchron. Daher kann es eine Verzögerung zwischen dem Ablaufdatum und dem Datum geben, an dem Amazon S3 ein Objekt entfernt. S3 Storage Lens enthält keine Metriken für Objekte, die abgelaufen sind, aber nicht entfernt wurden. Weitere Informationen über die Ablaufaktionen im S3-Lebenszyklus finden Sie unter Auslaufende Objekte.

Metrikglossar von S3 Storage Lens

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Gesamtspe icher	StorageBytes	Der Gesamtspe icher, einschlie ßlich unvollstä ndiger mehrteili ger Uploads, Objektmetadaten und Löschmark ierungen	Kost i	Übe	N	-
Anzahl Objekte	ObjectCount	Die Gesamtzahl der Objekte	Kost i	Übe	N	-
Durchschn ittliche Objektgröße	-	Die durchschn ittliche Objektgrö ße	Kost i	Übe	Y	sum(Stora geBytes)/

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
						sum(Objec tCount)
Aktive Buckets	-	Die Gesamtzah l der Buckets in aktiver Nutzung mit Speicher > 0 Bytes	Kost i	Übe	Y	-
Buckets	-	Die Gesamtanzahl der Buckets	Kost i	Übe	Y	-
Konten	-	Die Anzahl der Konten, deren Speicher zum Bereich gehören	Kost i	Übe	Y	-
Aktuelle Versions-Bytes	CurrentVe rsionStor ageBytes	Die Anzahl der Bytes, die eine aktuelle Version eines Objekts sind	Kost i	Kost imie	N	-
% Bytes der aktuellen Version	-	Der Prozentsa tz der Bytes im Bereich, die eine aktuelle Version sind	Kost i	Kost imie	Y	sum(Curre ntVersion StorageBy tes)/ sum(StorageBy tes)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Anzahl der Objekte der aktuellen Version	CurrentVersionObjectCount	Die Anzahl der aktuellen Versionsobjekte	Kost i	Kost imie	N	-
% Objekte der aktuellen Version	-	Der Prozentsatz der Objekte im Bereich, die eine aktuelle Version sind	Kost i	Kost imie	Y	sum(CurrentVersionObjectCount)/sum(ObjectCount)
Nicht aktuelle Versions-Bytes	NonCurrentVersionStorageBytes	Die Anzahl der Bytes der nicht aktuellen Version	Kost i	Kost imie	N	-
% Bytes der nicht aktuellen Version	-	Der Prozentsatz der Bytes im Bereich, die nicht aktuelle Versionen sind	Kost i	Kost imie	Y	sum(NonCurrentVersionStorageBytes)/sum(StorageBytes)
Anzahl nicht aktueller Versionsobjekte	NonCurrentVersionObjectCount	Die Anzahl der Objekte der nicht aktuellen Versionen	Kost i	Kost imie	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% Objekte der nicht aktuellen Version	-	Der Prozentsatz der Objekte im Bereich, die eine nicht aktuelle Version sind	Kosti	Kostimier	Y	sum(NonCurrentVersionObjectCount)/ sum(ObjectCount)
Löschmarkierungsbytes	DeleteMarkerStorageBytes	Die Anzahl der Bytes im Bereich, die Löschmarkierungen sind	Kosti	Kostimier	N	-
% Löschmarkierungsbytes	-	Der Prozentsatz der Bytes im Bereich, die Löschmarkierungen sind	Kosti	Kostimier	Y	sum(DeleteMarkerStorageBytes)/ sum(StorageBytes)
Anzahl der Löschmarkierungsobjekte	DeleteMarkerObjectCount	Die Gesamtzahl der Objekte mit einer Löschmarkierung	Kosti	Kostimier	N	-
% Objekte mit Löschmarkierung	-	Der Prozentsatz der Objekte im Bereich mit einer Löschmarkierung	Kosti	Kostimier	Y	sum(DeleteMarkerObjectCount)/ sum(ObjectCount)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Unvollstä ndige Bytes für mehrteilige Uploads	Incomplet eMultipar tUploadSt orageBytes	Die Gesamtzah l der Bytes im Bereich für unvollständige mehrteilige Uploads	Kost i	Kost imie	N	-
% unvollstä ndige Bytes für mehrteilige Uploads	-	Der Prozentsa tz der Bytes im Bereich, die das Ergebnis unvollstä ndiger mehrteiliger Uploads sind	Kost i	Kost imie	Y	sum(Incom pleteMult ipartUplo adStorage Bytes)/ su m(Storage Bytes)
Anzahl unvollständiger mehrteiliger Uploads	Incomplet eMultipar tUploadOb jectCount	Die Anzahl der Objekte im Bereich, die unvollständige mehrteilige Uploads sind	Kost i	Kost imie	N	-
% unvollstä ndige mehrteili ge Uploads	-	Der Prozentsa tz der Objekte im Bereich, die unvollständige mehrteilige Uploads sind	Kost i	Kost imie	Y	sum(Incom pleteMult ipartUplo adObjectC ount)/ sum (ObjectCo unt)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Unvollständige mehrteilige Upload-Speicherbytes, die älter als 7 Tage sind	IncompleteMPUStorageBytesOlderThan7Days	Die Gesamtzahl der Bytes im Bereich für unvollständige mehrteilige Uploads, die älter als 7 Tage sind	Kosti	Kostimier	N	-
% unvollständige mehrteilige Upload-Speicherbytes, die älter als 7 Tage sind	-	Die Gesamtzahl der Bytes für unvollständige mehrteilige Uploads, die älter als 7 Tage sind	Kosti	Kostimier	Y	sum(IncompleteMPUStorageBytesOlderThan7Days)/sum(StorageBytes)
Anzahl der unvollständigen mehrteiligen Upload-Objekte, die älter als 7 Tage sind	IncompleteMPUObjectCountOlderThan7Days	Die Anzahl der Objekte, die unvollständige mehrteilige Uploads und älter als 7 Tage sind	Kosti	Kostimier	N	-
% der Anzahl der unvollständigen mehrteiligen Upload-Objekte, die älter als 7 Tage sind	-	Die Prozentsatz der Objekte, die unvollständige mehrteilige Uploads und älter als 7 Tage sind	Kosti	Kostimier	Y	sum(IncompleteMPUObjectCountOlderThan7Days)/sum(ObjectCount)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Anzahl der Lebenszyk lusregeln für den Übergang	TransitionLifecycleRuleCount	Die Anzahl der Lebenszyk lusregeln für die Übertragung von Objekten in eine andere Speicherk lasse	Adva	Kost imie	N	-
Durchschn ittliche Lebenszyk lusregeln für den Übergang pro Bucket	-	Die durchschn ittliche Anzahl der Lebenszyk lusregeln für die Übertragung von Objekten in eine andere Speicherk lasse	Adva	Kost imie	Y	sum(TransitionLifecycleRuleCount)/ sum(DistinctNumberOfBuckets)
Anzahl der Lebenszyk lusregeln für den Ablauf	ExpirationLifecycleRuleCount	Die Anzahl der Lebenszyk lusregeln für das Ablaufen von Objekten	Adva	Kost imie	N	-
Durchschn ittliche Lebenszyk lusregeln für den Ablauf pro Bucket	-	Die durchschn ittliche Anzahl der Lebenszyk lusregeln für das Ablaufen von Objekten	Adva	Kost imie	Y	sum(ExpirationLifecycleRuleCount)/ sum(DistinctNumberOfBuckets)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Anzahl der Lebenszyk lusregeln für den Übergang nicht aktueller Versionen	Noncurren tVersionT ransition Lifecycle RuleCount	Die Anzahl der Lebenszyk lusregeln für die Übertragung von nicht aktuellen Objektversionen in eine andere Speicherklasse	Adva	Kost imie	N	
Durchschn ittliche Lebenszyk lusregeln für den Übergang von nicht aktuellen Objektver sionen pro Bucket	-	Die durchschn ittliche Anzahl der Lebenszyk lusregeln für die Übertragung von nicht aktuellen Objektversionen in eine andere Speicherklasse	Adva	Kost imie	Y	sum(Noncu rrentVers ionTransi tionLifec ycleRuleC ount)/ sum (Distinct NumberOfB uckets)
Anzahl der Lebenszyk lusregeln für den Ablauf nicht aktueller Versionen	Noncurren tVersionE xpiration Lifecycle RuleCount	Die Anzahl der Lebenszyk lusregeln für das Ablaufen von nicht aktuellen Objektversionen	Adva	Kost imie	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Durchschn ittliche Lebenszyk lusregeln für den Ablauf von nicht aktuellen Objektver sionen pro Bucket	-	Die durchschn ittliche Anzahl der Lebenszyk lusregeln für das Ablaufen von nicht aktuellen Objektversionen	Adva	Kost imie	Y	sum(Noncu rrentVers ionExpira tionLifec ycleRuleC ount)/ sum (Distinct NumberOfB uckets)
Anzahl der Lebenszyk lusregeln zum Abbrechen unvollständiger mehrteiliger Uploads	AbortInco mpleteMPU Lifecycle RuleCount	Die Anzahl der Lebenszyk lusregeln zum Löschen unvollstä ndiger mehrteiliger Uploads	Adva	Kost imie	N	-
Die durchschn ittliche Anzahl von Lebenszyk lusregeln zum Abbruch unvollständiger mehrteiliger Uploads pro Bucket	-	Die durchschn ittliche Anzahl der Lebenszyk lusregeln zum Löschen unvollstä ndiger mehrteiliger Uploads	Adva	Kost imie	Y	sum(Abort Incomplet eMPULifec ycleRuleC ount)/ sum (Distinct NumberOfB uckets)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Anzahl der Lebenszyk lusregeln für abgelaufene Objekte mit Löschmark ierung	ExpiredOb jectDelet eMarkerLi fecycleRu leCount	Anzahl der Lebenszyk lusregeln zum Entfernen abgelaufener Löschmark ierungen für Objekte	Adva	Kost imie	N	-
Durchschn ittliche Anzahl der Lebenszyk lusregeln zum Entfernen abgelaufener Löschmark ierungen für Objekte	-	Die durchschn ittliche Anzahl der Lebenszyk lusregeln zum Entfernen abgelaufener Löschmark ierungen für Objekte	Adva	Kost imie	Y	sum(Expir edObjectD eleteMark erLifecyc leRuleCou nt)/ sum(D istinctNu mberOfBuc kets)
Anzahl der gesamten Lebenszyk lusregeln	TotalLife cycleRuleCount	Die Gesamtzah l der Lebenszyk lusregeln	Adva	Kost imie	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Durchschn ittliche Anzahl von Lebenszyk lusregeln pro Bucket	-	Die durchschn ittliche Anzahl der Lebenszyk lusregeln	Adva	Kost imie	Y	sum(Total LifecycleRuleCount)/ sum(Dis tinctNumb erOfBucke ts)
Verschlüsselte Bytes	Encrypted StorageBytes	Die Gesamtzah l der verschlüs selten Bytes	Kost i	Date tz	N	-
% verschlüs selte Bytes	-	Der Prozentsatz der Gesamtzah l von Bytes, die verschlüsselt sind	Kost i	Date tz	Y	sum(Encry ptedObjec tCount)/ s um(Storag eBytes)
Anzahl der verschlüsselten Objekte	Encrypted ObjectCount	Die Gesamtzahl der Objekte, die verschlüsselt sind	Kost i	Date tz	N	-
% verschlüs selte Objekte	-	Der Prozentsatz der Objekte, die verschlüsselt sind	Kost i	Date tz	Y	sum(Encry ptedStora geBytes)/ sum(Objec tCount)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Unverschl üsselte Bytes	UnencryptedStorageBytes	Die Anzahl der unverschlüsselten Bytes	Kosti	Datetz	Y	sum(StorageBytes) - sum(EncryptedStorageBytes)
% unverschl üsselte Bytes	-	Der Prozentsatz der Bytes, die unverschlüsselt sind	Kosti	Datetz	Y	sum(UnencryptedStorageBytes)/ sum(StorageBytes)
Unverschl üsselte Objektanzahl	UnencryptedObjectCount	Die Gesamtzahl der Objekte, die unverschlüsselt sind	Kosti	Datetz	Y	sum(ObjectCount) - sum(EncryptedObjectCount)
% unverschl üsselte Objekte	-	Der Prozentsatz der unverschl üsselten Objekte	Kosti	Datetz	Y	sum(UnencryptedStorageBytes)/ sum(ObjectCount)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Replizierte Speicher-Bytes-Quelle	ReplicatedStorageBytesSource	Die Gesamtzahl der Bytes, die aus dem Quell-Bucket repliziert werden	Kosti	Datetz	N	-
% replizierte Bytes-Quelle	-	Der Prozentsatz der Gesamtzahl der Bytes, die aus dem Quell-Bucket repliziert werden	Kosti	Datetz	Y	sum(ReplicatedStorageBytesSource)/ sum(StorageBytes)
Quelle der Anzahl der replizierten Objekte	ReplicatedObjectCountSource	Die Anzahl der replizierten Objekte aus dem Quell-Bucket	Kosti	Datetz	N	-
% replizierte Objektquelle	-	Der Prozentsatz der Gesamtzahl der Objekte, die aus dem Quell-Bucket repliziert werden	Kosti	Datetz	Y	sum(ReplicatedStorageObjectCount)/ sum(ObjectCount)
Ziel der Replikationsspeicher-Bytes	ReplicatedStorageBytes	Die Gesamtzahl der Bytes, die in den Ziel-Bucket repliziert werden	Kosti	Datetz	Y	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% repliziertes Bytes-Ziel	-	Der Prozentsatz der Gesamtzahl der Bytes, die in den Ziel-Bucket repliziert werden	Kost i	Date tz	Y	sum(Repli catedStor ageBytes) / sum(Stor ageBytes)
Ziel der Anzahl der replizierten Objekte	Replicate dObjectCount	Die Anzahl der Objekte, die in den Ziel-Bucket repliziert werden	Kost i	Date tz	Y	-
% Ziel replizier ter Objekte	-	Der Prozentsatz der Gesamtzahl der Objekte, die in den Ziel-Bucket repliziert werden	Kost i	Date tz	Y	sum(Repli catedObje ctCount)/ sum(Objec tCount)
Bytes mit Objektsperre	ObjectLoc kEnableds torageBytes	Die Gesamtzah l der Speicherb ytes mit aktivierter Objektsperre	Kost i	Date tz	Y	sum(Unenc ryptedSto rageBytes)/ sum(Obj ectLockEn abledStor ageCount) - sum(Obje ctLockEna bledStora geBytes)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% Bytes mit Objektsperre	-	Der Prozentsatz der Speicherbytes mit aktivierter Objektsperre	Kosti	Datetz	Y	sum(ObjectLockEnabledStorageBytes)/sum(StorageBytes)
Anzahl der Objekte mit aktivierter Objektsperre	ObjectLockEnabledObjectCount	Die Gesamtzahl der Objekte mit Objektsperre	Kosti	Datetz	Y	-
% Objekte mit Objektsperre	-	Der Prozentsatz der Gesamtzahl der Objekte mit aktivierter Objektsperre	Kosti	Datetz	Y	sum(ObjectLockEnabledObjectCount)/sum(ObjectCount)
Bucket-Anzahl mit aktiviertem Versioning	VersioningEnabledBucketCount	Die Anzahl der Buckets, für die S3 Versioning aktiviert ist	Kosti	Datetz	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% Buckets mit aktiviertem Versioning	-	Der Prozentsatz der Buckets, für die S3 Versioning aktiviert ist	Kost i	Date tz	Y	sum(Versi oningEnab ledBucket Count)/ su m(Distinc tNumberOf Buckets)
Anzahl der MFA-lösch aktivierten Buckets	MFADelete EnabledBu cketCount	Die Anzahl der Buckets, für die das Löschen von MFA (Multi-Factor Authentication) aktiviert ist	Kost i	Date tz	N	-
% der MFA-löschaktivierten Buckets	-	Der Prozentsatz der Buckets, für die das Löschen von MFA (Multi-Factor Authentic ation) aktiviert ist	Kost i	Date tz	Y	sum(MFADe leteEnabl edBucketC ount)/ sum (Distinct NumberOfB uckets)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
SSE-KMS-fähige Bucket-Anzahl	SSEKMSEnabledBucketCount	Die Anzahl der Buckets, die serverseitige Verschlüsselung mit AWS Key Managemen t Service-S chlüsseln (SSE-KMS) als Bucket-Standardver schlüsselung verwenden	Kost i	Date tz	N	-
% der SSE-KMS-fähigen Buckets	-	Der Prozentsa tz der Buckets, die SSE-KMS als Bucket-Standardver schlüsselung verwenden	Kost i	Date tz	Y	sum(SSEKMSEnabledBucketCount)/ sum(DistinctNumberOfBuckets)
Alle nicht unterstützten Signatura nforderungen	AllUnsupportedSignatureRequests	Die Gesamtzah l der Anforderu ngen, die nicht unterstützte AWS-Signaturversionen verwenden	Adva	Date tz	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% aller nicht unterstützten Signatura nforderungen	-	Der Prozentsa tz der Anforderu ngen, die nicht unterstützte AWS-Signaturversionen verwenden	Adva	Date tz	Y	sum(AllUn supported Signature Requests) / sum(AllR equests)
Alle nicht unterstützten TLS-Anfor derungen	AllUnsupp ortedTLSR equests	Die Anzahl der Anforderungen, die nicht unterstüt zte Transport Layer Security (TLS)-Versionen verwenden	Adva	Date tz	N	-
% aller nicht unterstützten TLS-Anfor derungen	-	Der Prozentsa tz der Anforderu ngen, die nicht unterstützte TLS-Signaturversionen verwenden	Adva	Date tz	Y	sum(AllUn supported TLSReques ts)/ sum(A llRequest s)
Alle SSE-KMS-Anforderungen	AllSSEKMS Requests	Die Gesamtzah l der Anforderu ngen, die SSE-KMS spezifizieren	Adva	Date tz	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% aller SSE-KMS-Anforderungen	-	Der Prozentsatz der Anforderungen, die SSE-KMS spezifizieren	Adva	Datentz	Y	sum(AllSSEKMSRequests)/sum(AllRequests)
Anzahl der Replikationsregeln für dieselbe Region	SameRegionReplicationRuleCount	Die Anzahl der Replikationsregeln für Replikation innerhalb derselben Region (SRR)	Adva	Datentz	N	-
Durchschnittliche Replikationsregeln für Replikation innerhalb derselben Region	-	Die durchschnittliche Anzahl von Replikationsregeln für SRR	Adva	Datentz	Y	sum(SameRegionReplicationRuleCount)/sum(DistinctNumberOfBuckets)
Anzahl der regionsübergreifenden Replikationsregeln	CrossRegionReplicationRuleCount	Die Anzahl der Replikationsregeln für regionsübergreifende Replikation (CRR)	Adva	Datentz	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Durchschn ittliche Replikati onsregeln für regionsüb ergreifende Replikation pro Bucket	-	Die durchschn ittliche Anzahl von Replikationsregeln für CRR	Adva	Date tz	Y	sum(Cross RegionRep licationR uleCount) / sum(Dist inctNumbe rOfBucket s)
Anzahl der Replikati onsregeln für dasselbe Konto	SameAccou ntReplica tionRuleCount	Die Anzahl der Replikationsregeln für die Replikati on innerhalb desselben Kontos	Adva	Date tz	N	-
Durchschn ittliche Replikati onsregeln für Replikati on innerhalb desselben Kontos pro Bucket	-	Die durchschn ittliche Anzahl der Replikationsregeln für die Replikati on innerhalb desselben Kontos	Adva	Date tz	Y	sum(SameA ccountRep licationR uleCount) / sum(Dist inctNumbe rOfBucket s)
Kontenübe rgreifend e Replikati onsregelanzahl	CrossAcco untReplic ationRuleCount	Die Anzahl der Replikationsregeln für die kontoüber greifende Replikati on	Adva	Date tz	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Durchschn ittliche Replikati onsregeln für die kontoüber greifende Replikation pro Bucket	-	Die durchschn ittliche Anzahl der Replikationsregeln für die kontoüber greifende Replikati on	Adva	Date tz	Y	sum(Cross AccountRe plication RuleCount)/ sum(Dis tinctNumb erOfBucke ts)
Ungültige Anzahl der Zielrepli kationsregeln	InvalidDe stination Replicati onRuleCount	Die Anzahl der Replikati onsregeln mit einem Replikati onsziel, das nicht gültig ist	Adva	Date tz	N	-
Durchschn ittliche ungültige Anzahl der Zielrepli kationsregeln	-	Die durchschn ittliche Anzahl der Replikati onsregeln mit einem Replikati onsziel, das nicht gültig ist	Adva	Date tz	Y	sum(Inval idReplica tionRuleC ount)/ sum (Distinct NumberOfB uckets)
Gesamte Replikati onsregeln anzahl	-	Die Gesamtzah l der Replikati onsregeln	Adva	Date tz	Y	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Durchschn ittliche Anzahl der Replikati onsregeln pro Bucket	-	Die durchschn ittliche Gesamtzah l der Replikati onsregeln	Adva	Date tz	Y	sum(all replicati on rule count metrics)/ sum(Disti nctNumber OfBuckets)
Anzahl der Buckets mit Bucket-Ei gentümer erzwungen für Objekteig entümerschaft	ObjectOwn ershipBuc ketOwnerE nforcedBu cketCount	Die Gesamtzah l der Buckets, für die Zugriffss teuerungslisten (ACLs) deaktivie rt sind, indem die Einstellu ng „Bucket-Eigentümer erzwungen“ für die Objekteig entümerschaft verwendet wird	Kost i	Zugr erwa	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% Bucket-Eigentümer erzwungen für Objekteigentümerschaft	-	Der Prozentsatz der Buckets, für die ACLs deaktiviert sind, indem die Einstellung „Bucket-Eigentümer erzwungen“ für die Objekteigentümerschaft verwendet wird	Kosti	Zugrerwa	Y	sum(ObjectOwnershipBucketOwnerEnforcedBucketCount)/ sum (DistinctNumberOfBuckets)
Anzahl der Buckets mit Bucket-Eigentümer bevorzugt für Objekteigentümerschaft	ObjectOwnershipBucketOwnerPreferredBucketCount	Die Gesamtzahl der Buckets, die die Einstellung „Bucket-Eigentümer bevorzugt“ für die Objekteigentümerschaft verwenden	Kosti	Zugrerwa	N	-
% der Buckets mit Bucket-Eigentümer bevorzugt für Objekteigentümerschaft	-	Der Prozentsatz der Buckets, die die Einstellung „Bucket-Eigentümer bevorzugt“ für die Objekteigentümerschaft verwenden	Kosti	Zugrerwa	Y	sum(ObjectOwnershipBucketOwnerPreferredBucketCount)/ sum(DistinctNumberOfBuckets)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Objekt-Autor Bucket-Anzahl des Objekt-Ei gentümers	ObjectOwn ershipObj ectWriter BucketCount	Die Gesamtzahl der Buckets, die die Einstellung „Objektschreiber“ für die Objekteig entümerschaft verwenden	Kost i	Zugr erwa	N	-
% der Buckets mit Objektsch reiber für Objekteig entümerschaft	-	Der Prozentsatz der Buckets, die die Einstellung „Objektschreiber“ für die Objekteig entümerschaft verwenden	Kost i	Zugr erwa	Y	sum(Objec tOwnershi pObjectWr iterBucke tCount)/ s um(Distin ctNumberO fBuckets)
Anzahl aktiviert er Buckets für Transfer Acceleration	TransferA ccelerati onEnabled BucketCount	Die Gesamtzah l der Buckets, für die Transfer Acceleration aktiviert ist	Kost i	Leist	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% der aktiviert en Buckets für Transfer Acceleration	-	Der Prozentsa tz der Buckets, für die Transfer Acceleration aktiviert ist	Kost i	Leist	Y	sum(Trans ferAccele rationEna bledBucke tCount)/ s um(Distin ctNumberO fBuckets)
Anzahl der aktivierten Buckets für Ereignisb enachrich tigungen	EventNoti ficationE nabledBuc ketCount	Die Gesamtzah l der Buckets, für die Ereignisb enachrichtigungen aktiviert sind	Kost i	Ereig e	N	
% der aktiviert en Buckets für Ereignisb enachrich tigungen	-	Der Prozentsa tz der Buckets, für die Ereignisb enachrichtigungen aktiviert sind	Kost i	Ereig e	Y	sum(Event Notificat ionEnable dBucketCo unt)/ sum(DistinctN umberOfBu ckets)
Alle Anforderu ngen	AllRequests	Die Gesamtzahl der gesendeten - Anforderungen	Adva	Aktiv	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
GET-Anfor derungen	GetRequests	Die Gesamtzah l der gesendete n GET-Anforder ungen	Adva	Aktiv	N	-
PUT-Anfor derungen	PutRequests	Die Gesamtzah l der gesendete n PUT-Anforder ungen	Adva	Aktiv	N	-
HEAD-Anfo rderungen	HeadRequests	Die Gesamtzah l der gesendete n HEAD-Anforder ungen	Adva	Aktiv	N	-
DELETE-An forderungen	DeleteReq uests	Die Gesamtzah l der gesendeten DELETE-Anforder ungen	Adva	Aktiv	N	-
Listenanf orderungen	ListRequests	Die Gesamtzah l der gesendete n LIST-Anforder ungen	Adva	Aktiv	N	-
POST-Anfo rderungen	PostRequests	Die Gesamtzah l der gesendete n POST-Anforder ungen	Adva	Aktiv	N	-
SELECT-An forderungen	SelectRequests	Die Gesamtzahl der S3-SELECT-Anforderungen	Adva	Aktiv	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Ausgewählte gescannte Bytes	SelectScannedBytes	Die Anzahl der gescannten S3-SELECT-Bytes	Adva	Aktiv	N	-
Ausgewählte zurückgegebene Bytes	SelectReturnedBytes	Die Anzahl der zurückgegebenen S3-SELECT-Bytes	Adva	Aktiv	N	-
Heruntergeladene Bytes	BytesDownloaded	Die Anzahl der Bytes, die heruntergeladen wurden	Adva	Aktiv	N	-
% Abrufrate	-	Der Prozentsatz der Bytes, die heruntergeladen wurden	Adva	Aktiv	Y	sum(BytesDownloaded)/sum(StorageBytes)
Hochgeladene Bytes	BytesUploaded	Die Anzahl der hochgeladenen Bytes	Adva	Aktiv	N	-
% Aufnahmeverhältnis	-	Der Prozentsatz der Bytes, die hochgeladen wurden	Adva	Aktiv	Y	sum(BytesUploaded)/sum(StorageBytes)
4xx-Fehler	4xxErrors	Die Gesamtzahl der HTTP-4xx-Statuscodes	Adva	Aktiv	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
5xx-Fehler	5xxErrors	Die Gesamtzahl der HTTP-5xx-Statuscodes	Adva	Aktiv	N	-
Gesamtfehler	-	Die Summe aller 4xx- und 5xx-Fehler	Adva	Aktiv	Y	sum(4xxErrors) + sum(5xxErrors)
% Fehlerrate	-	Die Gesamtzahl der 4xx- und 5xx-Fehler als Prozentsatz der Gesamtanforderungen	Adva	Aktiv	Y	sum(TotalErrors)/ sum(TotalRequests)
200 OK Statusanzahl	200OKStatusCount	Die Gesamtzahl der 200-OK-Statuscodes	Adva	Deta rter Statu e	N	-
% Status 200 OK	-	Die Gesamtzahl der 200-OK-Statuscodes als Prozentsatz der Gesamtzahl der Anforderungen	Adva	Deta rter Statu e	Y	sum(200OKStatusCount)/ sum(AllRequests)
Statusanzahl 206 unvollständige Inhalte	206PartialContentStatusCount	Die Gesamtzahl der Statuscodes 206 für unvollständige Inhalte	Adva	Deta rter Statu e	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% Status 206 unvollständige Inhalte	-	Die Gesamtzahl der Statuscodes 206 für unvollständige Inhalte als Prozentsatz der Gesamtzahl der Anforderungen	Adva	Deta rter Statu e	Y	sum(206PartialContentStatusCount)/ su m(AllRequests)
Anzahl der Fehler 400 Bad Request	400BadRequestErrorCount	Die Gesamtzahl der Statuscodes 400 Bad Request	Adva	Deta rter Statu e	N	-
% Fehler 400 Bad Request	-	Die Gesamtzahl der Statuscodes 400 Bad Request als Prozentsatz der Gesamtzahl der Anforderungen	Adva	Deta rter Statu e	Y	sum(400BadRequestErrorCount)/ sum(All Requests)
Anzahl der Fehler 403 Forbidden	403ForbiddenErrorCount	Die Gesamtzahl der Statuscodes 403 Forbidden	Adva	Deta rter Statu e	N	-
% Fehler 403 Forbidden	-	Die Gesamtzahl der Statuscodes 403 Forbidden als Prozentsatz der Gesamtzahl der Anforderungen	Adva	Deta rter Statu e	Y	sum(403ForbiddenErrorCount) / sum(AllRequests)

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
Anzahl der Fehler 404 Not Found	404NotFoundErrorCount	Die Gesamtzahl der Statuscodes 404 Not Found	Adva	Deta rter Statu e	N	-
% Fehler 404 Not Found	-	Die Gesamtzahl der Statuscodes 404 Not Foundals Prozentsatz der Gesamtzahl der Anforderungen	Adva	Deta rter Statu e	Y	sum(404NotFoundErrorCount)/ sum(AllRequests)
Fehleranzahl 500 Internal Server Error	500InternalServerErrorCount	Die Gesamtzahl der Statuscodes 500 Internal Server Error	Adva	Deta rter Statu e	N	-
% Fehler 500 Internal Server Error	-	Die Gesamtzahl der Statuscodes 500 Internal Server Error als Prozentsatz der Gesamtzahl der Anforderungen	Adva	Deta rter Statu e	Y	sum(500InternalServerErrorCount)/ sum (AllRequests)
Anzahl der Fehler 503 Service Unavailable	503ServiceUnavailableErrorCount	Die Gesamtzahl der Statuscodes 503 Service Unavailable	Adva	Deta rter Statu e	N	-

Metrikname	CloudWatch und Export	Beschreibung	Stufe [1]	Kate [2]	Ab t	Abge te Metri mel
% Fehler 503 Service Unavailable	-	Die Gesamtzah l der Statuscod es 503 Service Unavailable als Prozentsatz der Gesamtzahl der Anforderungen	Adva	Deta rter Statu e	Y	sum(503Se rviceUnav ailableEr rorCount) / sum(AllR equests)

[1] Alle Metriken für das kostenlose Speicherkontingent sind auf Storage-Lens-Gruppenebene verfügbar. Metriken für die Advanced-Stufe sind auf Storage-Lens-Gruppenebene nicht verfügbar.

[2] Metriken für die Anzahl der Regeln und Metriken für Bucket-Einstellungen sind auf Präfixebene nicht verfügbar.

Arbeiten mit Amazon S3 Storage Lens unter Verwendung der Konsole und der API

Amazon S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Sie können Metriken von S3 Storage Lens verwenden, um zusammenfassende Erkenntnisse zu gewinnen und z. B. herauszufinden, wie viel Speicher Sie in Ihrer gesamten Organisation haben oder welche Buckets und Präfixe am schnellsten wachsen. Außerdem können Sie anhand der Metriken von S3 Storage Lens umfassende Möglichkeiten zur Kostenoptimierung aufdecken, bewährte Methoden für den Datenschutz implementieren und die Leistung von Anwendungs-Workloads verbessern. Sie können beispielsweise Buckets identifizieren, für die keine S3-Lebenszyklusregeln gelten, damit unvollständige mehrteilige Uploads, die älter als 7 Tage sind, ablaufen. Sie können auch Buckets identifizieren, die nicht den bewährten Datenschutzmethoden entsprechen, z. B. die Verwendung von S3 Replication oder S3 Versionierung. S3 Storage Lens analysiert Metriken, um kontextbezogene Empfehlungen zur Optimierung der Speicherkosten und zur Anwendung bewährter Datenschutzmethoden zu geben.

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.

Dieser Abschnitt enthält Beispiele für das Erstellen, Aktualisieren und Anzeigen von S3-Storage-Lens-Konfigurationen und das Ausführen von funktionsbezogenen Vorgängen. Auch Anwendungsfälle für den Einsatz von S3 Storage Lens mit AWS Organizations werden in diesen Beispielen abgedeckt. Ersetzen Sie alle Variablenwerte in den Beispielen durch Ihre eigenen Werte.

Themen

Verwenden von Amazon S3 Storage Lens in der Konsole

Amazon S3 Storage Lens ist eine Cloud-Speicheranalysefunktion, mit der Sie unternehmensweite Einblicke in die Nutzung und Aktivität von Objektspeichern erhalten können. Sie können Metriken von S3 Storage Lens verwenden, um zusammenfassende Erkenntnisse zu gewinnen und z. B. herauszufinden, wie viel Speicher Sie in Ihrer gesamten Organisation haben oder welche Buckets und Präfixe am schnellsten wachsen. Außerdem können Sie anhand der Metriken von S3 Storage Lens umfassende Möglichkeiten zur Kostenoptimierung aufdecken, bewährte Methoden für den Datenschutz implementieren und die Leistung von Anwendungs-Workloads verbessern. Sie können beispielsweise Buckets identifizieren, für die keine S3-Lebenszyklusregeln gelten, damit unvollständige mehrteilige Uploads, die älter als 7 Tage sind, ablaufen. Sie können auch Buckets identifizieren, die nicht den bewährten Datenschutzmethoden entsprechen, z. B. die Verwendung von S3 Replication oder S3 Versionierung. S3 Storage Lens analysiert Metriken, um kontextbezogene Empfehlungen zur Optimierung der Speicherkosten und zur Anwendung bewährter Datenschutzmethoden zu geben.

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet

außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Objekt- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.

> **Note**
>
> Es kann bis zu 48 Stunden dauern, bis Änderungen an der Dashboard-Konfiguration genau angezeigt bzw. visualisiert werden.

Themen

Erstellen und Aktualisieren von Amazon S3-Storage-Lens-Dashboards

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Objekt- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.

Das Standard-Dashboard von Amazon S3 Storage Lens ist default-account-dashboard. Dieses Dashboard ist von Amazon S3 vorkonfiguriert, damit Sie zusammengefasste Erkenntnisse und Trends der aggregierten kostenlosen und erweiterten Metriken Ihres gesamten Kontos in der Konsole visualisieren können. Sie können den Konfigurationsbereich des Standard-Dashboards nicht ändern, können aber die Metrikauswahl von den kostenlosen Metriken auf die kostenpflichtigen erweiterten Metriken und Empfehlungen upgraden, den optionalen Metrikexport konfigurieren oder sogar das Standard-Dashboard deaktivieren. Das Standard-Dashboard kann nicht gelöscht werden.

Sie können auch zusätzliche benutzerdefinierte S3-Storage-Lens-Dashboards erstellen, die auf Ihre Organisation in AWS Organizations oder auf bestimmte Regionen oder Buckets innerhalb eines Kontos beschränkt werden können.

Erstellen eines Amazon S3-Storage-Lens-Dashboards

Verwenden Sie die folgenden Schritte, um ein Amazon S3-Storage-Lens-Dashboard auf der Amazon S3-Konsole zu erstellen.

Schritt 1: Festlegen des Dashboard-Umfangs

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich unter S3 Storage Lens die Option Dashboards aus.
3. Klicken Sie auf Create dashboard (Dashboard erstellen).
4. Führen Sie auf der Seite Dashboard im Abschnitt Allgemein die folgenden Schritte aus:

   a. Geben Sie einen Dashboard-Namen ein.

      Dashboard-Namen müssen weniger als 65 Zeichen lang sein und dürfen keine Sonderzeichen oder Leerzeichen enthalten.

      > **Note**
      >
      > Nach dem Erstellen des Dashboards kann der Name nicht mehr geändert werden.

   b. Wählen Sie die Heimatregion für Ihr Dashboard aus. Die Dashboard-Metriken für alle Regionen in diesem Dashboard-Bereich werden zentral in dieser ausgewählten Heimatregion gespeichert.

   c. Sie haben die Wahl, Ihrem Dashboard Markierungen hinzuzufügen. Sie können Markierungen verwenden, um Berechtigungen für Ihr Dashboard zu verwalten und die Kosten für S3 Storage Lens nachzuverfolgen.

      Weitere Informationen finden Sie unter Steuern des Zugriffs mit Ressourcentags im IAM-Benutzerhandbuch und unter AWS-generierte Kostenzuordnungstags im AWS Billing - Benutzerhandbuch.

> **Note**
>
> Sie können Ihrer Dashboard-Konfiguration bis zu 50 Markierungen hinzufügen.

5. Führen Sie im Abschnitt Dashboard-Bereich die folgenden Schritte aus:

   a. Wählen Sie die Regionen und Buckets aus, die S3 Storage Lens im Dashboard ein- oder ausschließen soll.

   b. Wählen Sie die Buckets in den ausgewählten Regionen aus, die S3 Storage Lens ein- oder ausschließen soll. Sie können Buckets entweder ein- oder ausschließen. Beides ist nicht möglich. Diese Option ist nicht verfügbar, wenn Sie Dashboards auf Organisationsebene erstellen.

> **Note**
>
> - Sie können Regionen und Buckets entweder ein- oder ausschließen. Diese Option ist nur auf Regionen beschränkt, wenn Sie Dashboards auf Organisationsebene für alle Mitgliedskonten in Ihrer Organisation erstellen.
> - Sie können bis zu 50 Buckets auswählen, die Sie einschließen oder ausschließen möchten.

Schritt 2: Konfigurieren der Metrikauswahl

1. Wählen Sie im Abschnitt Metrikauswahl den Metriktyp aus, den Sie für dieses Dashboard aggregieren möchten.

   - Wählen Sie Free metrics (Kostenlose Metriken) aus, um kostenlose Metriken einzuschließen, die auf Bucket-Ebene aggregiert und 14 Tage lang für Abfragen verfügbar sind.
   - Wählen Sie Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) aus, um erweiterte Metriken und andere erweiterte Optionen zu aktivieren. Zu diesen Optionen gehören erweiterte Präfixaggregation, Amazon- CloudWatch Veröffentlichung und kontextbezogene Empfehlungen. Daten stehen für Abfragen für 15 Monate zur Verfügung. Erweiterte Metriken und Empfehlungen sind mit zusätzlichen Kosten verbunden. Weitere Informationen finden Sie unter Amazon S3 – Preise.

Weitere Informationen zu erweiterten und zu kostenlosen Metriken finden Sie unter Metrikauswahl.

2. Wählen Sie unter Advanced metrics and recommendations features (Erweiterte Metrik- und Empfehlungsfunktionen) die Optionen aus, Sie aktivieren möchten:

   - Advanced metrics (Erweiterte Metriken)
   - CloudWatch Veröffentlichen
   - Präfixzusammenfassung

   > **Important**
   >
   > Wenn Sie die Präfixaggregation für Ihre S3-Storage-Lens-Konfiguration aktivieren, werden Metriken auf Präfixebene nicht in veröffentlicht CloudWatch. Nur S3-Storage-Lens-Metriken auf Bucket-, Konto- und Organisationsebene werden in veröffentlicht CloudWatch.

3. Wenn Sie Advanced Metrics (Erweiterte Metriken) aktiviert haben, wählen Sie die Advanced metrics categories (Erweiterte Metrikkategorien) aus, die Sie in Ihrem S3-Storage-Lens-Dashboard anzeigen möchten:

   - Metriken für Aktivitäten
   - Detailed status code metrics (Detaillierte Statuscode-Metriken)
   - Advanced cost optimization metrics (Erweiterte Kostenoptimierungsmetriken)
   - Advanced data protection metrics (Erweiterte Datensicherheitsmetriken)

   Weitere Informationen zu den Metrikkategorien finden Sie unter Metrikkategorien. Eine vollständige Liste der Metriken finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

4. Wenn Sie die Präfixaggregation aktivieren möchten, konfigurieren Sie Folgendes:

   a. Wählen Sie die Mindestgröße für den Präfixschwellenwert für dieses Dashboard aus.

      Ein Präfixschwellenwert von 5 Prozent gibt beispielsweise an, dass Präfixe, die einen Anteil des gesamten Bucket-Speichers von 5 Prozent oder mehr ausmachen, aggregiert werden.

   b. Wählen Sie die Präfixtiefe aus.

Diese Einstellung gibt die maximale Anzahl von Ebenen an, bis zu denen die Präfixe ausgewertet werden. Die Präfixtiefe muss weniger als 10 betragen.

c. Geben Sie ein Präfixtrennzeichen ein.

Mit diesem Wert werden die einzelnen Präfixebenen identifiziert. Der Standardwert in Amazon S3 ist das Zeichen `/`, in Ihrer Speicherstruktur können jedoch ggf. andere Trennzeichen verwendet werden.

(Optional) Schritt 3: Exportieren von Metriken für das Dashboard

1. Wählen Sie im Abschnitt Metrics export (Metrik-Export) die Option Enable (Aktivieren) aus, um einen Metrik-Export zu erstellen, der jeden Tag in einem Ziel-Bucket Ihrer Wahl abgelegt wird.

   Der Metrik-Export liegt im CSV- oder Apache Parquet-Format vor. Er enthält den gleichen Datenumfang wie Ihr S3-Storage-Lens-Dashboard ohne die Empfehlungen.

2. Wenn Sie den Metrik-Export aktiviert haben, wählen Sie das Ausgabeformat für den täglichen Metrik-Export aus: CSV oder Apache Parquet.

   Parquet ist ein Open-Source-Dateiformat für Hadoop, das verschachtelte Daten in einem flachen Spaltenformat speichert.

3. Wählen Sie den S3-Ziel-Bucket für den Export Ihrer Metriken aus.

   Sie können einen Bucket im aktuellen Konto des S3-Storage-Lens-Dashboards auswählen. Oder Sie können eine andere auswählen AWS-Konto , wenn Sie über die Ziel-Bucket-Berechtigungen und die Konto-ID des Ziel-Bucket-Eigentümers verfügen.

4. Wählen Sie den S3-Ziel-Bucket (Format: `s3://`*`bucket-name`*`/`*`prefix`*) aus.

   Der Bucket muss sich in der Heimatregion Ihres S3-Storage-Lens-Dashboards befinden. In der S3-Konsole wird das Feld Destination bucket permission (Ziel-Bucket-Berechtigung) mit der Berechtigung angezeigt, die der Ziel-Bucket-Richtlinie von Amazon S3 hinzugefügt wird. Amazon S3 aktualisiert die Bucket-Richtlinie für den Ziel-Bucket, damit S3 Daten darin ablegen kann.

5. (Optional) Wenn Sie die serverseitige Verschlüsselung für Ihren Metriken aktivieren möchten, wählen Sie Specify an encryption key (Einen Verschlüsselungsschlüssel angeben) aus. Wählen Sie anschließend den Verschlüsselungstyp aus: Von Amazon S3 verwaltete Schlüssel (SSE-S3) oder AWS Key Management Service -Schlüssel (SSE-KMS).

Sie können entweder einen von Amazon S3 verwalteten Schlüssel (SSE-S3) und einen AWS Key Management Service (AWS KMS)-Schlüssel (SSE-KMS) auswählen.

6. (Optional) Um einen - AWS KMS Schlüssel anzugeben, müssen Sie einen KMS-Schlüssel auswählen oder einen Amazon-Ressourcennamen (ARN) eingeben.

   Wenn Sie einen kundenverwalteten Schlüssel auswählen, müssen Sie S3 Storage Lens in der AWS KMS -Schlüsselrichtlinie die Berechtigung zum Verschlüsseln erteilen. Weitere Informationen finden Sie unter Verwenden eines AWS KMS key zur Verschlüsselung von Metrik-Exporten.

7. Klicken Sie auf Dashboard erstellen.

Um sich einen besseren Überblick über Ihren Speicher zu verschaffen, können Sie eine oder mehrere S3-Storage-Lens-Gruppen erstellen und diese an das Dashboard anhängen. Eine Storage-Lens-Gruppe ist ein benutzerdefinierter Filter für Objekte, der auf Präfixen, Suffixen, Objekt-Tags, Objektgröße, Objektalter oder einer Kombination dieser Filter basiert.

Sie können S3-Storage-Lens-Gruppen verwenden, um detaillierte Einblicke in große freigegebene Buckets wie Data Lakes zu erhalten und fundiertere Geschäftsentscheidungen zu treffen. Sie können beispielsweise die Speicherzuweisung rationalisieren und die Erstellung von Kostenberichten optimieren, indem Sie die Speichernutzung für einzelne Projekte und Kostenstellen innerhalb eines oder mehrerer Buckets nach bestimmten Objektgruppen aufschlüsseln.

Zur Nutzung von S3-Storage-Lens-Gruppen müssen Sie das Dashboard aktualisieren, um erweiterte Metriken und Empfehlungen zu verwenden. Weitere Informationen zur Verwendung von S3-Storage-Lens-Gruppen finden Sie unter the section called “Arbeiten mit S3-Storage-Lens-Gruppen”.

Aktualisieren eines Amazon S3-Storage-Lens-Dashboards

Verwenden Sie die folgenden Schritte, um ein Amazon S3-Storage-Lens-Dashboard auf der Amazon S3-Konsole zu aktualisieren.

Schritt 1: Aktualisieren des Dashboard-Umfangs

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens, Dashboards aus.
3. Wählen Sie das Dashboard aus, das Sie bearbeiten möchten, und wählen Sie dann Bearbeiten.

Die Seite Edit dashboard (Dashboard bearbeiten) wird geöffnet.

> **Note**
>
> Folgendes können Sie nicht ändern:
>
> - Den Dashboard-Namen
> - Die Heimatregion
> - Der Bereich des Standard-Dashboards, das für den Speicher Ihres gesamten Kontos eingerichtet ist

4. (Optional) Auf der Dashboard-Konfigurationsseite im Abschnitt General (Allgemein) können Sie Ihr Dashboard aktualisieren und Markierungen hinzufügen.

   Sie können Markierungen verwenden, um Berechtigungen für Ihr Dashboard zu verwalten und die Kosten für S3 Storage Lens nachzuverfolgen. Weitere Informationen finden Sie unter Steuern des Zugriffs mit Ressourcentags im IAM-Benutzerhandbuch und unter AWS-generierte Kostenzuordnungstags im AWS Billing -Benutzerhandbuch.

   > **Note**
   >
   > Sie können Ihrer Dashboard-Konfiguration bis zu 50 Markierungen hinzufügen.

5. Führen Sie im Abschnitt Dashboard-Bereich die folgenden Schritte aus:

   a. Aktualisieren Sie die Regionen und Buckets, die S3 Storage Lens im Dashboard ein- oder ausschließen soll.

      > **Note**
      >
      > - Sie können Regionen und Buckets entweder ein- oder ausschließen. Diese Option ist nur auf Regionen beschränkt, wenn Sie Dashboards auf Organisationsebene für alle Mitgliedskonten in Ihrer Organisation erstellen.
      > - Sie können bis zu 50 Buckets auswählen, die Sie einschließen oder ausschließen möchten.

b. Aktualisieren Sie die Buckets in den ausgewählten Regionen, die S3 Storage Lens ein- oder ausschließen soll. Sie können Buckets entweder ein- oder ausschließen. Beides ist nicht möglich. Diese Option ist nicht vorhanden, wenn Sie Dashboards auf Organisationsebene erstellen.

Schritt 2: Aktualisieren der Metrikauswahl

1. Wählen Sie im Abschnitt Metrikauswahl den Metriktyp aus, den Sie für dieses Dashboard aggregieren möchten.

   - Wählen Sie Free metrics (Kostenlose Metriken) aus, um kostenlose Metriken einzuschließen, die auf Bucket-Ebene aggregiert und 14 Tage lang für Abfragen verfügbar sind.
   - Wählen Sie Advanced metrics and recommendations (Erweiterte Metriken und Empfehlungen) aus, um erweiterte Metriken und andere erweiterte Optionen zu aktivieren. Zu diesen Optionen gehören erweiterte Präfixaggregation, Amazon- CloudWatch Veröffentlichung und kontextbezogene Empfehlungen. Daten stehen für Abfragen für 15 Monate zur Verfügung. Erweiterte Metriken und Empfehlungen sind mit zusätzlichen Kosten verbunden. Weitere Informationen finden Sie unter Amazon S3 – Preise.

     Weitere Informationen zu erweiterten und zu kostenlosen Metriken finden Sie unter Metrikauswahl.

2. Wählen Sie unter Advanced metrics and recommendations features (Erweiterte Metrik- und Empfehlungsfunktionen) die Optionen aus, Sie aktivieren möchten:

   - Advanced metrics (Erweiterte Metriken)
   - CloudWatch Veröffentlichen
   - Präfixzusammenfassung

   > **Important**
   >
   > Wenn Sie die Präfixaggregation für Ihre S3-Storage-Lens-Konfiguration aktivieren, werden Metriken auf Präfixebene nicht in veröffentlicht CloudWatch. Nur S3-Storage-Lens-Metriken auf Bucket-, Konto- und Organisationsebene werden in veröffentlicht CloudWatch.

3. Wenn Sie Advanced Metrics (Erweiterte Metriken) aktiviert haben, wählen Sie die Advanced metrics categories (Erweiterte Metrikkategorien) aus, die Sie in Ihrem S3-Storage-Lens-Dashboard anzeigen möchten:

   - Metriken für Aktivitäten
   - Detailed status code metrics (Detaillierte Statuscode-Metriken)
   - Advanced cost optimization metrics (Erweiterte Kostenoptimierungsmetriken)
   - Advanced data protection metrics (Erweiterte Datensicherheitsmetriken)

   Weitere Informationen zu Metrikkategorien finden Sie unter Metrikkategorien. Eine vollständige Liste der Metriken finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

4. Wenn Sie die Präfixaggregation aktivieren möchten, konfigurieren Sie Folgendes:

   a. Wählen Sie die Mindestgröße für den Präfixschwellenwert für dieses Dashboard aus.

      Ein Präfixschwellenwert von 5 Prozent gibt beispielsweise an, dass Präfixe, die einen Anteil des gesamten Bucket-Speichers von 5 Prozent oder mehr ausmachen, aggregiert werden.

   b. Wählen Sie die Präfixtiefe aus.

      Diese Einstellung gibt die maximale Anzahl von Ebenen an, bis zu denen die Präfixe ausgewertet werden. Die Präfixtiefe muss weniger als 10 betragen.

   c. Geben Sie ein Präfixtrennzeichen ein.

      Dies ist der Wert, mit dem die einzelnen Präfixebenen identifiziert werden. Der Standardwert in Amazon S3 ist das Zeichen `/`, in Ihrer Speicherstruktur können jedoch ggf. andere Trennzeichen verwendet werden.

(Optional) Schritt 3: Exportieren von Metriken für das Dashboard

1. Wählen Sie im Abschnitt Metrics export (Metrik-Export) die Option Enable (Aktivieren) aus, um einen Metrik-Export zu erstellen, der jeden Tag in einem Ziel-Bucket Ihrer Wahl abgelegt wird. Wenn Sie den Metrik-Export deaktivieren möchten, wählen Sie Disable (Deaktivieren) aus.

   Der Metrik-Export liegt im CSV- oder Apache Parquet-Format vor. Er enthält den gleichen Datenumfang wie Ihr S3-Storage-Lens-Dashboard ohne die Empfehlungen.

2. Wenn diese Option aktiviert ist, wählen Sie das Ausgabeformat für den täglichen Metrik-Export aus: CSV oder Apache Parquet.

Parquet ist ein Open-Source-Dateiformat für Hadoop, das verschachtelte Daten in einem flachen Spaltenformat speichert.

3. Wählen Sie den S3-Ziel-Bucket für den Export Ihrer Metriken aus.

   Sie können einen Bucket im aktuellen Konto des S3-Storage-Lens-Dashboards auswählen. Oder Sie können eine andere auswählen AWS-Konto , wenn Sie über die Ziel-Bucket-Berechtigungen und die Konto-ID des Ziel-Bucket-Eigentümers verfügen.

4. Wählen Sie den S3-Ziel-Bucket (Format: `s3://bucket-name/prefix`) aus.

   Der Bucket muss sich in der Heimatregion Ihres S3-Storage-Lens-Dashboards befinden. In der S3-Konsole wird das Feld Destination bucket permission (Ziel-Bucket-Berechtigung) mit der Berechtigung angezeigt, die der Ziel-Bucket-Richtlinie von Amazon S3 hinzugefügt wird. Amazon S3 aktualisiert die Bucket-Richtlinie für den Ziel-Bucket, damit S3 Daten darin ablegen kann.

5. (Optional) Wenn Sie die serverseitige Verschlüsselung für Ihren Metriken aktivieren möchten, wählen Sie Specify an encryption key (Einen Verschlüsselungsschlüssel angeben) aus. Wählen Sie anschließend den Verschlüsselungstyp aus: Von Amazon S3 verwaltete Schlüssel (SSE-S3) oder AWS Key Management Service -Schlüssel (SSE-KMS).

   Sie können entweder einen von Amazon S3 verwalteten Schlüssel (SSE-S3) und einen AWS Key Management Service (AWS KMS)-Schlüssel (SSE-KMS) auswählen.

6. (Optional) Um einen - AWS KMS Schlüssel anzugeben, müssen Sie einen KMS-Schlüssel auswählen oder einen Amazon-Ressourcennamen (ARN) eingeben. Geben Sie unter AWS KMS -Schlüssel Ihren KMS-Schlüssel auf eine der folgenden Arten an:

   - Wenn Sie aus einer Liste verfügbarer KMS-Schlüssel auswählen möchten, wählen Sie Aus Ihren AWS KMS keys wählen und anschließend den KMS-Schlüssel in der Liste der verfügbaren Schlüssel aus.

     Sowohl die Von AWS verwalteter Schlüssel (`aws/s3`) als auch Ihre vom Kunden verwalteten Schlüssel werden in dieser Liste angezeigt. Weitere Informationen über vom Kunden verwaltete Schlüssel finden Sie unter Kundenschlüssel und AWS -Schlüssel im Entwicklerhandbuch zu AWS Key Management Service .

> **Note**
>
> Die Von AWS verwalteter Schlüssel (`aws/S3`) wird für die SSE-KMS-Verschlüsselung mit S3 Storage Lens nicht unterstützt.

- Wählen Sie zum Eingeben des KMS-Schlüssel-ARN AWS KMS key -ARN eingeben aus und geben Sie Ihren KMS-Schlüssel-ARN in das angezeigte Feld ein.
- Um einen neuen kundenverwalteten Schlüssel in der AWS KMS Konsole zu erstellen, wählen Sie KMS-Schlüssel erstellen aus.

  Wenn Sie einen kundenverwalteten Schlüssel auswählen, müssen Sie S3 Storage Lens in der AWS KMS -Schlüsselrichtlinie die Berechtigung zum Verschlüsseln erteilen. Weitere Informationen finden Sie unter Verwenden eines AWS KMS key zur Verschlüsselung von Metrik-Exporten.

  Weitere Informationen zum Erstellen eines AWS KMS key finden Sie unter Erstellen von Schlüsseln im AWS Key Management Service -Entwicklerhandbuch.

7. Wählen Sie Änderungen speichern aus.

Um sich einen besseren Überblick über Ihren Speicher zu verschaffen, können Sie eine oder mehrere S3-Storage-Lens-Gruppen erstellen und diese an das Dashboard anhängen. Eine Storage-Lens-Gruppe ist ein benutzerdefinierter Filter für Objekte, der auf Präfixen, Suffixen, Objekt-Tags, Objektgröße, Objektalter oder einer Kombination dieser Filter basiert.

Sie können S3-Storage-Lens-Gruppen verwenden, um detaillierte Einblicke in große freigegebene Buckets wie Data Lakes zu erhalten und fundiertere Geschäftsentscheidungen zu treffen. Sie können beispielsweise die Speicherzuweisung rationalisieren und die Erstellung von Kostenberichten optimieren, indem Sie die Speichernutzung für einzelne Projekte und Kostenstellen innerhalb eines oder mehrerer Buckets nach bestimmten Objektgruppen aufschlüsseln.

Zur Nutzung von S3-Storage-Lens-Gruppen müssen Sie das Dashboard aktualisieren, um erweiterte Metriken und Empfehlungen zu verwenden. Weitere Informationen zur Verwendung von S3-Storage-Lens-Gruppen finden Sie unter the section called “Arbeiten mit S3-Storage-Lens-Gruppen”.

Deaktivieren oder Löschen von Amazon S3-Storage-Lens-Dashboards

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet

außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Objekt- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.

Das Standard-Dashboard von Amazon S3 Storage Lens ist default-account-dashboard. Dieses Dashboard ist von Amazon S3 vorkonfiguriert, damit Sie zusammengefasste Erkenntnisse und Trends der aggregierten kostenlosen und erweiterten Metriken Ihres gesamten Kontos in der Konsole visualisieren können. Sie können den Konfigurationsbereich des Standard-Dashboards nicht ändern, können aber die Metrikauswahl von den kostenlosen Metriken auf die kostenpflichtigen erweiterten Metriken und Empfehlungen upgraden, den optionalen Metrikexport konfigurieren oder sogar das Standard-Dashboard deaktivieren. Das Standard-Dashboard kann nicht gelöscht werden.

Sie können ein Amazon S3-Storage-Lens-Dashboard in der Amazon S3-Konsole löschen oder deaktivieren. Durch das Deaktivieren oder Löschen eines Dashboards wird verhindert, dass es weitere Metriken generiert. Ein deaktiviertes Dashboard behält seine Konfigurationsinformationen, sodass es bei erneuter Aktivierung einfach wieder genutzt werden kann. Ein deaktiviertes Dashboard behält seine Verlaufsdaten bei, bis es nicht mehr für Abfragen verfügbar ist.

Die Daten für die Auswahl der kostenlosen Metriken stehen 14 Tage lang für Abfragen zur Verfügung. Die Daten für die Auswahl der erweiterten Metriken und Empfehlungen stehen 15 Monate für Abfragen zur Verfügung.

Deaktivieren eines Amazon S3-Storage-Lens-Dashboards

So deaktivieren Sie ein S3-Storage-Lens-Dashboard

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie deaktivieren möchten, und gehen Sie dann oben in der Liste auf Deaktivieren.
4. Bestätigen Sie auf der Bestätigungsseite, dass Sie das Dashboard deaktivieren möchten, indem Sie den Namen des Dashboards in das Textfeld eingeben und dann Bestätigen auswählen.

Löschen eines Amazon S3-Storage-Lens-Dashboards

> **Note**
>
> Das Standard-Dashboard können Sie nicht löschen. Eine Deaktivierung ist jedoch möglich. Bevor Sie ein selbst erstelltes Dashboard löschen, sollten Sie Folgendes beachten:
>
> - Als Alternative zum Löschen können Sie das Dashboard deaktivieren, damit es in Zukunft wieder aktiviert werden kann. Weitere Informationen finden Sie unter Deaktivieren eines Amazon S3-Storage-Lens-Dashboards.
> - Durch das Löschen eines Dashboards werden alle damit verbundenen Konfigurationseinstellungen gelöscht.
> - Sobald Sie ein Dashboard löschen, können die historischen Metrikdaten nicht mehr abgerufen werden. Diese historischen Daten werden weiterhin für eine Dauer von 15 Monaten aufbewahrt. Wenn Sie wieder auf diese Daten zugreifen möchten, müssen Sie ein neues Dashboard mit dem gleichen Namen erstellen, dass sich in derselben Heimatregion befindet wie das gelöschte Dashboard.

So löschen Sie ein S3-Storage-Lens-Dashboard

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und dann Dashboards aus.
3. Wählen Sie in der Liste Dashboards das Dashboard aus, das Sie löschen möchten, und gehen Sie dann oben in der Liste auf Löschen.
4. Bestätigen Sie auf der Seite Dashboards löschen, dass Sie das Dashboard löschen möchten, indem Sie den Namen des Dashboards in das Textfeld eingeben. Wählen Sie dann Confirm (Bestätigen) aus.

Arbeiten mit AWS Organizations zum Erstellen von Dashboards auf Organisationsebene

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen,

um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Objekt- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden.

Das Standard-Dashboard von Amazon S3 Storage Lens ist default-account-dashboard. Dieses Dashboard ist von Amazon S3 vorkonfiguriert, damit Sie zusammengefasste Erkenntnisse und Trends der aggregierten kostenlosen und erweiterten Metriken Ihres gesamten Kontos in der Konsole visualisieren können. Sie können den Konfigurationsbereich des Standard-Dashboards nicht ändern, können aber die Metrikauswahl von den kostenlosen Metriken auf die kostenpflichtigen erweiterten Metriken und Empfehlungen upgraden, den optionalen Metrikexport konfigurieren oder sogar das Standard-Dashboard deaktivieren. Das Standard-Dashboard kann nicht gelöscht werden.

Sie können auch zusätzliche S3-Storage-Lens-Dashboards erstellen, die sich auf bestimmte AWS-Regionen, S3-Buckets oder andere AWS-Konten in Ihrer Organisation konzentrieren.

Ein S3-Storage-Lens-Dashboard bietet umfassende Informationen zum zugehörigen Speicherbereich. In einem Dashboard werden über 30 Metriken visualisiert, die Trends und andere Informationen darstellen, einschließlich Speicherzusammenfassung, Kosteneffizienz, Datenschutz und Aktivitäten.

Amazon S3 Storage Lens kann verwendet werden, um Speichermetriken und Nutzungsdaten für alle Konten zu sammeln, die Teil Ihrer AWS Organizations Hierarchie sind. Dazu müssen Sie verwenden AWS Organizations und den vertrauenswürdigen Zugriff von S3 Storage Lens mithilfe Ihres AWS Organizations Verwaltungskontos aktivieren.

Wenn der vertrauenswürdige Zugriff aktiviert ist, können Sie anderen Konten in Ihrer Organisation den delegierten Administratorzugriff gewähren. Diese Konten können dann organisationsweite Dashboards und Konfigurationen für S3 Storage Lens erstellen. Weitere Informationen zum Aktivieren des vertrauenswürdigen Zugriffs finden Sie in Amazon S3 Lens und AWS Organizations im AWS Organizations -Benutzerhandbuch.

Die folgenden Konsolensteuerelemente sind nur für die AWS Organizations Verwaltungskonten verfügbar.

Aktivieren des vertrauenswürdigen Zugriffs für S3 Storage Lens in Ihrer Organisation

Durch die Aktivierung des vertrauenswürdigen Zugriffs kann Amazon S3 Storage Lens über AWS Organizations API-Operationen auf Ihre AWS Organizations Hierarchie, Mitgliedschaft und Struktur zugreifen. S3 Storage Lens wird zu einem vertrauenswürdigen Service für die Struktur

Ihrer gesamten Organisation. Immer wenn eine Dashboard-Konfiguration erstellt wird, kann der Service in den Verwaltungskonten oder den delegierten Administratorkonten Ihrer Organisation serviceverknüpfte Rollen erstellen.

Die serviceverknüpfte Rolle gewährt S3 Storage Lens Berechtigungen, um Organisationen zu beschreiben, Konten aufzulisten, eine Liste des -Servicezugriffs für die Organisationen zu verifizieren und die delegierten Administratoren der Organisation abzurufen. Auf diese Weise kann S3 Storage Lens kontoübergreifende Speichernutzungs- und Aktivitätsmetriken für Dashboards in Konten Ihrer Organisation erfassen.

Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für Amazon S3 Storage Lens.

> **Note**
>
> - Vertrauenswürdiger Zugriff kann nur im Verwaltungskonto aktiviert werden.
> - S3-Storage-Lens-Dashboards oder -Konfigurationen für Ihre Organisation können nur über das Verwaltungskonto und von delegierten Administratoren erstellt werden.

So aktivieren Sie den vertrauenswürdigen Zugriff für S3 Storage Lens

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und Organization settings (Einstellungen für Organisation) aus.
3. Wählen Sie unter Organizations-Zugriff die Option Bearbeitenaus.

   Die Seite Organizations-Zugriff wird geöffnet. Hier können Sie den vertrauenswürdigen Zugriff für S3 Storage Lens aktivieren. Dadurch können Sie und alle anderen Kontoinhaber, die Sie als delegierte Administratoren hinzufügen, Dashboards für alle Konten und Speicher in Ihrer Organisation erstellen.

Deaktivieren des vertrauenswürdigen Zugriffs für S3 Storage Lens in Ihrer Organisation

Durch die Deaktivierung des vertrauenswürdigen Zugriffs wird die Funktionalität von S3 Storage Lens auf die Kontoebene beschränkt. Jeder Kontoinhaber sieht nur die S3-Storage-Lens-Vorteile, die auf

den Bereich seines Kontos beschränkt sind. Die Funktionen für die gesamte Organisation werden nicht angezeigt. Alle Dashboards, die einen vertrauenswürdigen Zugriff erfordern, werden nicht mehr aktualisiert. Die Dashboards können ihre historischen Daten jedoch gemäß dem jeweils geltenden Zeitraum, in dem Daten für Abfragen verfügbar sind abfragen.

Wenn der delegierte Administratorzugriff eines Kontos entfernt wird, kann der Kontoinhaber nur noch auf Kontoebene auf die Dashboard-Metriken von S3 Storage Lens zugreifen. Alle von ihnen erstellten Organisations-Dashboards werden nicht mehr aktualisiert, aber sie können ihre historischen Daten für den Zeitraum abfragen, in dem sie für Abfragen verfügbar sind.

> **Note**
>
> - Durch die Deaktivierung des vertrauenswürdigen Zugriffs werden auch automatisch alle Dashboards auf Organisationsebene deaktiviert, da S3 Storage Lens keinen vertrauenswürdigen Zugriff mehr auf die Organisationskonten hat, um Speichermetriken zu erfassen und zu aggregieren.
> - Die Verwaltungs- und Stellvertreter-Administratorkonten können weiterhin die historischen Daten für diese deaktivierten Dashboards sehen und diese Daten abfragen, solange sie verfügbar sind.

So deaktivieren Sie den vertrauenswürdigen Zugriff für S3 Storage Lens

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und Organization settings (Einstellungen für Organisation) aus.
3. Wählen Sie unter Organizations-Zugriff die Option Bearbeitenaus.

   Die Seite Organizations-Zugriff wird geöffnet. Hier können Sie den vertrauenswürdigen Zugriff für S3 Storage Lens deaktivieren.

Registrieren von delegierten Administratoren für S3 Storage Lens

Nachdem Sie den vertrauenswürdigen Zugriff aktiviert haben, können Sie andere Konten in Ihrer Organisation für den delegierten Administratorzugriff registrieren. Wenn ein Konto als delegierter Administrator registriert ist, erhält das Konto die Berechtigung für den Zugriff auf alle

schreibgeschützten AWS Organizations API-Operationen. Das macht die Mitglieder und Strukturen Ihrer Organisation sichtbar, sodass die delegierten Administratoren in Ihrem Namen S3-Storage-Lens-Dashboards erstellen können.

So registrieren Sie delegierte Administratoren für S3 Storage Lens

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Storage Lens und Organization settings (Einstellungen für Organisation) aus.
3. Wählen Sie im Abschnitt Delegierter Zugriff für Konten die Option Konto hinzufügen aus.

   Die Seite Delegierter Administratorzugriff wird geöffnet. Hier können Sie eine AWS-Konto -ID als delegierten Administrator hinzufügen, um Dashboards auf Organisationsebene für alle Konten und Speicher in Ihrer Organisation zu erstellen.

Aufheben der Registrierung delegierter Administratoren für S3 Storage Lens

Sie können den delegierten Administratorzugriff für Konten in Ihrer Organisation wieder entfernen. Wenn ein Konto als delegierter Administrator abgemeldet wird, verliert es die Berechtigung für den Zugriff auf alle schreibgeschützten AWS Organizations API-Operationen, die den Mitgliedern und Strukturen Ihrer Organisation Transparenz bieten.

> **Note**
>
> - Durch das Aufheben der Registrierung eines delegierten Administrators werden automatisch auch alle Dashboards auf Organisationsebene deaktiviert, die der delegierte Administrator erstellt hat.
> - Die delegierten Administratorkonten können weiterhin die historischen Daten für diese deaktivierten Dashboards gemäß dem jeweiligen Zeitraum anzeigen, in dem die Daten für Abfragen verfügbar sind.

So heben Sie die Registrierung von Konten für den delegierten Administratorzugriff auf

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie im linken Navigationsbereich Storage Lens und Organization settings (Einstellungen für Organisation) aus.
3. Wählen Sie im Abschnitt Konten mit delegiertem Zugriff die Konto-ID aus, deren Registrierung Sie aufheben möchten, und gehen Sie dann auf Entfernen.

Amazon S3-Storage-Lens-Beispiele für die Verwendung der AWS CLI.

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Objekt- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden. Weitere Informationen finden Sie unter Bewertung der Speicheraktivität und -nutzung mit Amazon S3 Storage Lens.

In den folgenden Beispielen wird gezeigt, wie Sie S3 Storage Lens mit der AWS Command Line Interface verwenden.

Themen

- Hilfsdateien für die Verwendung von Amazon S3 Storage Lens
- Verwenden von Amazon S3-Storage-Lens-Konfigurationen mit der AWS CLI
- Beispiele für das Verwenden von Amazon S3 Storage Lens mit AWS Organizations unter Verwendung der AWS CLI

Hilfsdateien für die Verwendung von Amazon S3 Storage Lens

Verwenden Sie die folgenden JSON-Dateien und die zugehörigen wichtigen Eingaben für Ihre Beispiele.

Beispiel für S3-Storage-Lens-Konfiguration in JSON

Example **`config.json`**

Die Datei `config.json` enthält Details einer S3-Storage-Lens-Konfiguration auf Organisationsebene mit erweiterten Metriken und Empfehlungen. Wenn Sie das folgende

Beispiel verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre eigenen Informationen.

> **Note**
>
> Für erweiterte Metriken und Empfehlungen fallen zusätzliche Gebühren an. Weitere Informationen finden Sie unter Erweiterte Metriken und Empfehlungen.

```
{
  "Id": "SampleS3StorageLensConfiguration", //Use this property to identify your S3
 Storage Lens configuration.
  "AwsOrg": { //Use this property when enabling S3 Storage Lens for AWS Organizations.
    "Arn": "arn:aws:organizations::123456789012:organization/o-abcdefgh"
  },
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "DetailedStatusCodesMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
        "IsEnabled":true
      },
      "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
      },
      "AdvancedCostOptimizationMetrics": {
        "IsEnabled":true
      },
      "DetailedStatusCodesMetrics": {
        "IsEnabled":true
      },
      "PrefixLevel":{
```

```
        "StorageMetrics":{
          "IsEnabled":true,
          "SelectionCriteria":{
            "MaxDepth":5,
            "MinStorageBytesPercentage":1.25,
            "Delimiter":"/"
          }
        }
      }
    }
  },
  "Exclude": { //Replace with "Include" if you prefer to include Regions.
    "Regions": [
      "eu-west-1"
    ],
    "Buckets": [ //This attribute is not supported for AWS Organizations-level
 configurations.
      "arn:aws:s3:::source_bucket1"
    ]
  },
  "IsEnabled": true, //Whether the configuration is enabled
  "DataExport": { //Details about the metrics export
    "S3BucketDestination": {
      "OutputSchemaVersion": "V_1",
      "Format": "CSV", //You can add "Parquet" if you prefer.
      "AccountId": "111122223333",
      "Arn": "arn:aws:s3:::destination-bucket-name", // The destination bucket for your
 metrics export must be in the same Region as your S3 Storage Lens configuration.
      "Prefix": "prefix-for-your-export-destination",
      "Encryption": {
        "SSES3": {}
      }
    },
    "CloudWatchMetrics": {
      "IsEnabled": true
    }
  }
}
```

Beispiel für S3-Storage-Lens-Konfiguration mit Storage-Lens-Gruppen in JSON

Example **`config.json`**

Die Datei `config.json` enthält die Details, die Sie bei Verwendung von Storage-Lens-Gruppen auf die Storage-Lens-Konfiguration anwenden möchten. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie die *`user input placeholders`* durch Ihre Informationen.

Um alle Storage-Lens-Gruppen an das Dashboard anzuhängen, aktualisieren Sie die Storage-Lens-Konfiguration mit der folgenden Syntax:

```
{
  "Id": "ExampleS3StorageLensConfiguration",
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
      "IsEnabled":true
      },
    "StorageLensGroupLevel": {},
  "IsEnabled": true
}
```

Verwenden Sie die folgende Syntax, wenn Sie nur zwei Storage-Lens-Gruppen (*`slg-1`* und *`slg-2`*) in die Storage-Lens-Dashboard-Konfiguration aufnehmen möchten:

```
{
  "Id": "ExampleS3StorageLensConfiguration",
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
```

```
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
      "IsEnabled":true
      },
   "StorageLensGroupLevel": {
        "SelectionCriteria": {
            "Include": [
                "arn:aws:s3:us-east-1:111122223333:storage-lens-group/slg-1",
                "arn:aws:s3:us-east-1:444455556666:storage-lens-group/slg-2"
            ]
    },
  "IsEnabled": true
}
```

Verwenden Sie die folgende Syntax, um bestimmte Storage-Lens-Gruppen vom Anhängen an die Dashboard-Konfiguration auszuschließen:

```
{
  "Id": "ExampleS3StorageLensConfiguration",
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
      "IsEnabled":true
      },
   "StorageLensGroupLevel": {
        "SelectionCriteria": {
            "Exclude": [
                "arn:aws:s3:us-east-1:111122223333:storage-lens-group/slg-1",
                "arn:aws:s3:us-east-1:444455556666:storage-lens-group/slg-2"
            ]
```

```
    },
  "IsEnabled": true
}
```

Beispiel für S3-Storage-Lens-Konfiguration mit Tags in JSON

Example **`tags.json`**

Die Datei `tags.json` enthält die Tags, die Sie auf Ihre S3-Storage-Lens-Konfiguration anwenden möchten. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
[
    {
        "Key": "key1",
        "Value": "value1"
    },
    {
        "Key": "key2",
        "Value": "value2"
    }
]
```

Beispiel für S3-Storage-Lens-Konfiguration für IAM Berechtigungen

Example **`permissions.json`** – Spezifischer Dashboard-Name

Diese Beispielrichtlinie zeigt eine IAM-Datei `permissions.json` von S3 Storage Lens mit einem bestimmten Dashboard-Namen. Ersetzen Sie *`value1`*, *`us-east-1`*, *`your-dashboard-name`* und *`example-account-id`* durch Ihre eigenen Werte.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetStorageLensConfiguration",
                "s3:DeleteStorageLensConfiguration",
                "s3:PutStorageLensConfiguration"
            ],
```

```
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/key1": "value1"
                }
            },
            "Resource": "arn:aws:s3:us-east-1:example-account-id:storage-lens/your-
dashboard-name"
        }
    ]
}
```

Example **`permissions.json`** – Kein spezifischer Dashboard-Name

Diese Beispielrichtlinie zeigt eine IAM-Datei `permissions.json` von S3 Storage Lens ohne einen bestimmten Dashboard-Namen. Ersetzen Sie *`value1`*, *`us-east-1`* und *`example-account-id`* durch Ihre eigenen Werte.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetStorageLensConfiguration",
                "s3:DeleteStorageLensConfiguration",
                "s3:PutStorageLensConfiguration"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/key1": "value1"
                }
            },
            "Resource": "arn:aws:s3:us-east-1:example-account-id:storage-lens/*"
        }
    ]
}
```

Verwenden von Amazon S3-Storage-Lens-Konfigurationen mit der AWS CLI

Sie können die AWS CLI verwenden, um S3-Storage-Lens-Konfigurationen aufzulisten, zu erstellen, zu löschen, abzurufen, zu markieren und zu aktualisieren. In den folgenden Beispielen werden die

JSON-Hilfsdateien für wichtige Eingaben verwendet. Wenn Sie diese Beispiele verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre Informationen.

Erstellen einer S3-Storage-Lens-Konfiguration

Example Erstellen einer S3-Storage-Lens-Konfiguration

```
aws s3control put-storage-lens-configuration --account-id=111122223333 --
config-id=example-dashboard-configuration-id --region=us-east-1 --storage-lens-
configuration=file://./config.json --tags=file://./tags.json
```

Erstellen einer S3-Storage-Lens-Konfiguration ohne Markierungen

Example Erstellen einer S3-Storage-Lens-Konfiguration ohne Markierungen

```
aws s3control put-storage-lens-configuration --account-id=222222222222 --config-
id=your-configuration-id --region=us-east-1 --storage-lens-configuration=file://./
config.json
```

Abrufen einer S3-Storage-Lens-Konfiguration

Example Abrufen einer S3-Storage-Lens-Konfiguration

```
aws s3control get-storage-lens-configuration --account-id=222222222222 --config-
id=your-configuration-id --region=us-east-1
```

Auflisten von S3-Storage-Lens-Konfigurationen ohne das nächste Token

Example Auflisten von S3-Storage-Lens-Konfigurationen ohne das nächste Token

```
aws s3control list-storage-lens-configurations --account-id=222222222222 --region=us-
east-1
```

Listet S3-Storage-Lens-Konfigurationen auf

Example Listet S3-Storage-Lens-Konfigurationen auf

```
aws s3control list-storage-lens-configurations --account-id=222222222222 --region=us-
east-1 --next-token=abcdefghij1234
```

Löschen einer S3-Storage-Lens-Konfiguration

Example Löschen einer S3-Storage-Lens-Konfiguration

```
aws s3control delete-storage-lens-configuration --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id
```

Hinzufügen von Markierungen zu einer S3-Storage-Lens-Konfiguration

Example Hinzufügen von Markierungen zu einer S3-Storage-Lens-Konfiguration

```
aws s3control put-storage-lens-configuration-tagging --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id --tags=file://./tags.json
```

Abrufen von Markierungen für eine S3-Storage-Lens-Konfiguration

Example Abrufen von Markierungen für eine S3-Storage-Lens-Konfiguration

```
aws s3control get-storage-lens-configuration-tagging --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id
```

Löschen von Markierungen für eine S3-Storage-Lens-Konfiguration

Example Löschen von Markierungen für eine S3-Storage-Lens-Konfiguration

```
aws s3control delete-storage-lens-configuration-tagging --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id
```

Beispiele für das Verwenden von Amazon S3 Storage Lens mit AWS Organizations unter Verwendung der AWS CLI

Verwenden Sie Amazon S3 Storage Lens, um Speichermetriken und Nutzungsdaten für alle Konten zu erfassen, die Teil Ihrer AWS Organizations-Hierarchie sind. Weitere Informationen finden Sie unter Using Amazon S3 Storage Lens with AWS Organizations (Verwenden von Amazon S3 Storage Lens mit AO).

Aktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

Example Aktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

```
aws organizations enable-aws-service-access --service-principal storage-
lens.s3.amazonaws.com
```

Deaktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

Example Deaktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

```
aws organizations disable-aws-service-access --service-principal storage-
lens.s3.amazonaws.com
```

Registrieren delegierter Organizations-Administratoren für S3 Storage Lens

Example Registrieren delegierter Organizations-Administratoren für S3 Storage Lens

Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *111122223333* durch die entsprechende AWS-Konto-ID.

```
aws organizations register-delegated-administrator --service-principal storage-
lens.s3.amazonaws.com --account-id 111122223333
```

Registrierung delegierter Organizations-Administratoren für S3 Storage Lens aufheben

Example Registrierung delegierter Organizations-Administratoren für S3 Storage Lens aufheben

Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *111122223333* durch die entsprechende AWS-Konto-ID.

```
aws organizations deregister-delegated-administrator --service-principal storage-
lens.s3.amazonaws.com --account-id 111122223333
```

Amazon-S3-Storage-Lens-Beispiele für die Verwendung des SDK for Java

S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um

Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden. Weitere Informationen finden Sie unter Bewertung der Speicheraktivität und -nutzung mit Amazon S3 Storage Lens.

In den folgenden Beispielen wird gezeigt, wie Sie S3 Storage Lens mit AWS SDK for Java verwenden.

Themen

- Verwenden von Amazon S3-Storage-Lens-Konfigurationen mit dem SDK for Java

Verwenden von Amazon S3-Storage-Lens-Konfigurationen mit dem SDK for Java

Sie können das SDK for Java verwenden, um S3-Storage-Lens-Konfigurationen aufzulisten, zu erstellen, abzurufen und zu aktualisieren. In den folgenden Beispielen werden die JSON-Hilfsdateien für wichtige Eingaben verwendet.

Themen

- Erstellen und Aktualisieren einer S3-Storage-Lens-Konfiguration
- Löschen einer S3-Storage-Lens-Konfiguration
- Abrufen einer S3-Storage-Lens-Konfiguration
- Listet S3-Storage-Lens-Konfigurationen auf
- Hinzufügen von Markierungen zu einer S3-Storage-Lens-Konfiguration
- Abrufen von Markierungen für eine S3-Storage-Lens-Konfiguration
- Löschen von Markierungen für eine S3-Storage-Lens-Konfiguration
- Aktualisieren der Standardkonfiguration von S3 Storage Lens mit erweiterten Metriken und Empfehlungen
- Anhängen einer Storage-Lens-Gruppe an ein S3-Storage-Lens-Dashboard
- Verwenden von Amazon S3 Storage Lens mit AWS Organizations-Beispielen unter Verwendung von SDK für Java

Erstellen und Aktualisieren einer S3-Storage-Lens-Konfiguration

Example Erstellen und Aktualisieren einer S3-Storage-Lens-Konfiguration

```
package aws.example.s3control;
```

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.ActivityMetrics;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.CloudWatchMetrics;
import com.amazonaws.services.s3control.model.Format;
import com.amazonaws.services.s3control.model.Include;
import com.amazonaws.services.s3control.model.OutputSchemaVersion;
import com.amazonaws.services.s3control.model.PrefixLevel;
import com.amazonaws.services.s3control.model.PrefixLevelStorageMetrics;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.S3BucketDestination;
import com.amazonaws.services.s3control.model.SSES3;
import com.amazonaws.services.s3control.model.SelectionCriteria;
import com.amazonaws.services.s3control.model.StorageLensAwsOrg;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensDataExport;
import com.amazonaws.services.s3control.model.StorageLensDataExportEncryption;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateAndUpdateDashboard {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        String exportAccountId = "Destination Account ID";
        String exportBucketArn = "arn:aws:s3:::destBucketName"; // The destination
 bucket for your metrics export must be in the same Region as your S3 Storage Lens
 configuration.
        String awsOrgARN = "arn:aws:organizations::123456789012:organization/o-
abcdefgh";
        Format exportFormat = Format.CSV;

        try {
```

```
            SelectionCriteria selectionCriteria = new SelectionCriteria()
                    .withDelimiter("/")
                    .withMaxDepth(5)
                    .withMinStorageBytesPercentage(10.0);
            PrefixLevelStorageMetrics prefixStorageMetrics = new
PrefixLevelStorageMetrics()
                    .withIsEnabled(true)
                    .withSelectionCriteria(selectionCriteria);
            BucketLevel bucketLevel = new BucketLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withAdvancedCostOptimizationMetrics(new
AdvancedCostOptimizationMetrics().withIsEnabled(true))
                    .withAdvancedDataProtectionMetrics(new
AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withPrefixLevel(new
PrefixLevel().withStorageMetrics(prefixStorageMetrics));
            AccountLevel accountLevel = new AccountLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withAdvancedCostOptimizationMetrics(new
AdvancedCostOptimizationMetrics().withIsEnabled(true))
                    .withAdvancedDataProtectionMetrics(new
AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withBucketLevel(bucketLevel);

            Include include = new Include()
                    .withBuckets(Arrays.asList("arn:aws:s3:::bucketName"))
                    .withRegions(Arrays.asList("us-west-2"));

            StorageLensDataExportEncryption exportEncryption = new
StorageLensDataExportEncryption()
                    .withSSES3(new SSES3());
            S3BucketDestination s3BucketDestination = new S3BucketDestination()
                    .withAccountId(exportAccountId)
                    .withArn(exportBucketArn)
                    .withEncryption(exportEncryption)
                    .withFormat(exportFormat)
                    .withOutputSchemaVersion(OutputSchemaVersion.V_1)
                    .withPrefix("Prefix");
            CloudWatchMetrics cloudWatchMetrics = new CloudWatchMetrics()
                    .withIsEnabled(true);
```

```
            StorageLensDataExport dataExport = new StorageLensDataExport()
                    .withCloudWatchMetrics(cloudWatchMetrics)
                    .withS3BucketDestination(s3BucketDestination);

            StorageLensAwsOrg awsOrg = new StorageLensAwsOrg()
                    .withArn(awsOrgARN);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withInclude(include)
                    .withDataExport(dataExport)
                    .withAwsOrg(awsOrg)
                    .withIsEnabled(true);

            List<StorageLensTag> tags = Arrays.asList(
                    new StorageLensTag().withKey("key-1").withValue("value-1"),
                    new StorageLensTag().withKey("key-2").withValue("value-2")
            );

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
                    .withTags(tags)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Löschen einer S3-Storage-Lens-Konfiguration

Example Löschen einer S3-Storage-Lens-Konfiguration

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.DeleteStorageLensConfigurationRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class DeleteDashboard {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.deleteStorageLensConfiguration(new
 DeleteStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
}
```

Abrufen einer S3-Storage-Lens-Konfiguration

Example Abrufen einer S3-Storage-Lens-Konfiguration

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.GetStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.GetStorageLensConfigurationResult;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class GetDashboard {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            final StorageLensConfiguration configuration =
                    s3ControlClient.getStorageLensConfiguration(new
 GetStorageLensConfigurationRequest()
                            .withAccountId(sourceAccountId)
                            .withConfigId(configurationId)
                    ).getStorageLensConfiguration();

            System.out.println(configuration.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
```

```
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Listet S3-Storage-Lens-Konfigurationen auf

Example Listet S3-Storage-Lens-Konfigurationen auf

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.ListStorageLensConfigurationEntry;
import com.amazonaws.services.s3control.model.ListStorageLensConfigurationsRequest;

import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class ListDashboard {

    public static void main(String[] args) {
        String sourceAccountId = "Source Account ID";
        String nextToken = "nextToken";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            final List<ListStorageLensConfigurationEntry> configurations =
                    s3ControlClient.listStorageLensConfigurations(new
 ListStorageLensConfigurationsRequest()
                            .withAccountId(sourceAccountId)
                            .withNextToken(nextToken)
                    ).getStorageLensConfigurationList();
```

```
            System.out.println(configurations.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Hinzufügen von Markierungen zu einer S3-Storage-Lens-Konfiguration

Example Hinzufügen von Markierungen zu einer S3-Storage-Lens-Konfiguration

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import
 com.amazonaws.services.s3control.model.PutStorageLensConfigurationTaggingRequest;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class PutDashboardTagging {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";

        try {
            List<StorageLensTag> tags = Arrays.asList(
                    new StorageLensTag().withKey("key-1").withValue("value-1"),
                    new StorageLensTag().withKey("key-2").withValue("value-2")
            );
```

```
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfigurationTagging(new
 PutStorageLensConfigurationTaggingRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withTags(tags)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Abrufen von Markierungen für eine S3-Storage-Lens-Konfiguration

Example Abrufen von Markierungen für eine S3-Storage-Lens-Konfiguration

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.DeleteStorageLensConfigurationRequest;
import
 com.amazonaws.services.s3control.model.GetStorageLensConfigurationTaggingRequest;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;
```

```
public class GetDashboardTagging {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            final List<StorageLensTag> s3Tags = s3ControlClient
                    .getStorageLensConfigurationTagging(new
 GetStorageLensConfigurationTaggingRequest()
                            .withAccountId(sourceAccountId)
                            .withConfigId(configurationId)
                    ).getTags();

            System.out.println(s3Tags.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Löschen von Markierungen für eine S3-Storage-Lens-Konfiguration

Example Löschen von Markierungen für eine S3-Storage-Lens-Konfiguration

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import
 com.amazonaws.services.s3control.model.DeleteStorageLensConfigurationTaggingRequest;
```

```
import static com.amazonaws.regions.Regions.US_WEST_2;

public class DeleteDashboardTagging {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.deleteStorageLensConfigurationTagging(new
 DeleteStorageLensConfigurationTaggingRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Aktualisieren der Standardkonfiguration von S3 Storage Lens mit erweiterten Metriken und Empfehlungen

Example Aktualisieren der Standardkonfiguration von S3 Storage Lens mit erweiterten Metriken und Empfehlungen

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
```

```
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.ActivityMetrics;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.Format;
import com.amazonaws.services.s3control.model.Include;
import com.amazonaws.services.s3control.model.OutputSchemaVersion;
import com.amazonaws.services.s3control.model.PrefixLevel;
import com.amazonaws.services.s3control.model.PrefixLevelStorageMetrics;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.S3BucketDestination;
import com.amazonaws.services.s3control.model.SSES3;
import com.amazonaws.services.s3control.model.SelectionCriteria;
import com.amazonaws.services.s3control.model.StorageLensAwsOrg;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensDataExport;
import com.amazonaws.services.s3control.model.StorageLensDataExportEncryption;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateDefaultConfigWithPaidFeatures {

    public static void main(String[] args) {
        String configurationId = "default-account-dashboard"; // This configuration ID
 cannot be modified.
        String sourceAccountId = "Source Account ID";

        try {
            SelectionCriteria selectionCriteria = new SelectionCriteria()
                    .withDelimiter("/")
                    .withMaxDepth(5)
                    .withMinStorageBytesPercentage(10.0);
            PrefixLevelStorageMetrics prefixStorageMetrics = new
 PrefixLevelStorageMetrics()
                    .withIsEnabled(true)
                    .withSelectionCriteria(selectionCriteria);
            BucketLevel bucketLevel = new BucketLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withPrefixLevel(new
 PrefixLevel().withStorageMetrics(prefixStorageMetrics));
```

```
            AccountLevel accountLevel = new AccountLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withBucketLevel(bucketLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

> **Note**
>
> Für erweiterte Metriken und Empfehlungen fallen zusätzliche Gebühren an. Weitere Informationen finden Sie unter Erweiterte Metriken und Empfehlungen.

Anhängen einer Storage-Lens-Gruppe an ein S3-Storage-Lens-Dashboard

Example Anhängen aller Storage-Lens-Gruppen an ein Dashboard

Im folgenden SDK-für-Java-Beispiel werden alle Storage-Lens-Gruppen im Konto *111122223333* an das Dashboard *DashBoardConfigurationId* angehängt.

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensGroupLevel;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateDashboardWithStorageLensGroups {
    public static void main(String[] args) {
        String configurationId = "ExampleDashboardConfigurationId";
        String sourceAccountId = "111122223333";

        try {
            StorageLensGroupLevel storageLensGroupLevel = new StorageLensGroupLevel();

            AccountLevel accountLevel = new AccountLevel()
                    .withBucketLevel(new BucketLevel())
                    .withStorageLensGroupLevel(storageLensGroupLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();
```

```
            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Anhängen von zwei Storage-Lens-Gruppen an ein Dashboard

Im folgenden AWS SDK for Java-Beispiel werden zwei Storage-Lens-Gruppen (*StorageLensGroupName1* und *StorageLensGroupName2*) an das Dashboard *ExampleDashboardConfigurationId* angehängt.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensGroupLevel;
import com.amazonaws.services.s3control.model.StorageLensGroupLevelSelectionCriteria;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateDashboardWith2StorageLensGroups {
    public static void main(String[] args) {
```

```
        String configurationId = "ExampleDashboardConfigurationId";
        String storageLensGroupName1 = "StorageLensGroupName1";
        String storageLensGroupName2 = "StorageLensGroupName2";
        String sourceAccountId = "111122223333";

        try {
            StorageLensGroupLevelSelectionCriteria selectionCriteria = new
StorageLensGroupLevelSelectionCriteria()
                    .withInclude(
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
+ ":storage-lens-group/" + storageLensGroupName1,
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
+ ":storage-lens-group/" + storageLensGroupName2);

            System.out.println(selectionCriteria);
            StorageLensGroupLevel storageLensGroupLevel = new StorageLensGroupLevel()
                    .withSelectionCriteria(selectionCriteria);

            AccountLevel accountLevel = new AccountLevel()
                    .withBucketLevel(new BucketLevel())
                    .withStorageLensGroupLevel(storageLensGroupLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
```

```
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Anhängen aller Storage-Lens-Gruppen mit Ausschlüssen

Im folgenden SDK-für-Java-Beispiel werden bis auf zwei angegebene Storage-Lens-Gruppen (*StorageLensGroupName1* und *StorageLensGroupName2*) alle Storage-Lens-Gruppen an das Dashboard *ExampleDashboardConfigurationId* angehängt.

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensGroupLevel;
import com.amazonaws.services.s3control.model.StorageLensGroupLevelSelectionCriteria;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateDashboardWith2StorageLensGroupsExcluded {
    public static void main(String[] args) {
        String configurationId = "ExampleDashboardConfigurationId";
        String storageLensGroupName1 = "StorageLensGroupName1";
        String storageLensGroupName2 = "StorageLensGroupName2";
        String sourceAccountId = "111122223333";

        try {
            StorageLensGroupLevelSelectionCriteria selectionCriteria = new
 StorageLensGroupLevelSelectionCriteria()
                    .withInclude(
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
 + ":storage-lens-group/" + storageLensGroupName1,
```

```
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
 + ":storage-lens-group/" + storageLensGroupName2);

            System.out.println(selectionCriteria);
            StorageLensGroupLevel storageLensGroupLevel = new StorageLensGroupLevel()
                    .withSelectionCriteria(selectionCriteria);

            AccountLevel accountLevel = new AccountLevel()
                    .withBucketLevel(new BucketLevel())
                    .withStorageLensGroupLevel(storageLensGroupLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwenden von Amazon S3 Storage Lens mit AWS Organizations-Beispielen unter Verwendung von SDK für Java

Verwenden Sie Amazon S3 Storage Lens, um Speichermetriken und Nutzungsdaten für alle Konten zu erfassen, die Teil Ihrer AWS Organizations-Hierarchie sind. Weitere Informationen finden Sie unter Using Amazon S3 Storage Lens with AWS Organizations (Verwenden von Amazon S3 Storage Lens mit AO).

Themen

Aktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

Example Aktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
import com.amazonaws.services.organizations.model.EnableAWSServiceAccessRequest;

public class EnableOrganizationsTrustedAccess {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            organizationsClient.enableAWSServiceAccess(new
 EnableAWSServiceAccessRequest()
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
```

```
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Deaktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

Example Deaktivieren des vertrauenswürdigen Zugriffs von Organizations für S3 Storage Lens

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
import com.amazonaws.services.organizations.model.DisableAWSServiceAccessRequest;

public class DisableOrganizationsTrustedAccess {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            // Make sure to remove any existing delegated administrator for S3 Storage
 Lens
            // before disabling access; otherwise, the request will fail.
            organizationsClient.disableAWSServiceAccess(new
 DisableAWSServiceAccessRequest()
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
        } catch (AmazonServiceException e) {
```

```
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Registrieren delegierter Organizations-Administratoren für S3 Storage Lens

Example Registrieren delegierter Organizations-Administratoren für S3 Storage Lens

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
import
 com.amazonaws.services.organizations.model.RegisterDelegatedAdministratorRequest;

public class RegisterOrganizationsDelegatedAdministrator {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            String delegatedAdminAccountId = "111122223333"; // Account Id for the
 delegated administrator.
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            organizationsClient.registerDelegatedAdministrator(new
 RegisterDelegatedAdministratorRequest()
                .withAccountId(delegatedAdminAccountId)
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
        } catch (AmazonServiceException e) {
```

```
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Registrierung delegierter Organizations-Administratoren für S3 Storage Lens aufheben

Example Registrierung delegierter Organizations-Administratoren für S3 Storage Lens aufheben

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
import
 com.amazonaws.services.organizations.model.DeregisterDelegatedAdministratorRequest;

public class DeregisterOrganizationsDelegatedAdministrator {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            String delegatedAdminAccountId = "111122223333"; // Account Id for the
 delegated administrator.
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            organizationsClient.deregisterDelegatedAdministrator(new
 DeregisterDelegatedAdministratorRequest()
                .withAccountId(delegatedAdminAccountId)
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
        } catch (AmazonServiceException e) {
```

```
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Arbeiten mit S3-Storage-Lens-Gruppen

Eine Amazon-S3-Storage-Lens-Gruppe aggregiert Metriken mithilfe benutzerdefinierter Filter auf der Grundlage von Objektmetadaten. Mit Storage-Lens-Gruppen können Sie Details zu den Eigenschaften Ihrer Daten anzeigen, etwa die Objektverteilung nach Alter, die gängigsten Dateitypen usw. Sie können Metriken beispielsweise nach Objekt-Tag filtern, um die am schnellsten wachsenden Datensätze zu ermitteln, oder den Speicher auf Grundlage der Objektgröße und des Objektalters visualisieren, um die Speicherarchivierungsstrategie festzulegen. Folglich vermitteln Amazon-S3-Storage-Lens-Gruppen Ihnen ein besseres Verständnis des S3-Speichers und unterstützen Sie bei seiner Optimierung.

Wenn Sie Storage-Lens-Gruppen verwenden, können Sie S3-Storage-Lens-Metriken anhand von Objektmetadaten wie Präfixen, Suffixen, Objekt-Tags, Objektgröße oder Objektalter analysieren und filtern. Diese Filter lassen sich auch kombinieren. Nachdem Sie eine Storage-Lens-Gruppe an das S3-Storage-Lens-Dashboard angehängt haben, können Sie die nach Amazon-S3-Storage-Lens-Gruppen aggregierten S3-Storage-Lens-Metriken direkt im Dashboard anzeigen.

Filtern Sie Messwerte beispielsweise auch nach Objektgröße oder Altersklassen, um festzustellen, welcher Speicheranteil aus kleinen Objekten besteht. Sie können diese Informationen dann mit S3 Intelligent-Tiering oder im S3-Lebenszyklus verwenden, um kleine Objekte zur Kosten- und Speicheroptimierung in verschiedene Speicherklassen zu überführen.

Themen

- Funktionsweise von S3-Storage-Lens-Gruppen
- Verwenden von Storage-Lens-Gruppen

Funktionsweise von S3-Storage-Lens-Gruppen

Mit Storage-Lens-Gruppen lassen sich Metriken mithilfe benutzerdefinierter Filter auf der Grundlage von Objektmetadaten aggregieren. Wenn Sie einen benutzerdefinierten Filter definieren, können Sie Präfixe, Suffixe, Objekt-Tags, Objektgrößen, Objektalter oder eine Kombination dieser benutzerdefinierten Filter verwenden. Bei der Erstellung einer Storage-Lens-Gruppe können Sie zudem einen einzelnen Filter oder mehrere Filterbedingungen angeben. Zur Angabe mehrerer Filterbedingungen verwenden Sie den logischen Operator `And` oder `Or`.

Wenn Sie eine Storage-Lens-Gruppe erstellen und konfigurieren, dient die Storage-Lens-Gruppe selbst als benutzerdefinierter Filter im Dashboard, an das Sie die Gruppe anhängen. Im Dashboard können Sie dann den Storage-Lens-Gruppenfilter verwenden, um Speichermetriken auf Grundlage des benutzerdefinierten Filters abzurufen, den Sie in der Gruppe definiert haben.

Um die Daten für die Storage-Lens-Gruppe im S3-Storage-Lens-Dashboard anzuzeigen, müssen Sie die Gruppe nach der Erstellung an das Dashboard anhängen. Nachdem die Storage-Lens-Gruppe an das Storage-Lens-Dashboard angehängt wurde, erfasst das Dashboard innerhalb von 48 Stunden Metriken zur Speichernutzung. Sie können diese Daten dann im Storage-Lens-Dashboard visualisieren oder sie über einen Metrikexport exportieren. Falls Sie vergessen, eine Storage-Lens-Gruppe an ein Dashboard anzuhängen, werden die Daten der Storage-Lens-Gruppe weder erfasst noch angezeigt.

> **Note**
>
> - Wenn Sie eine S3-Storage-Lens-Gruppe erstellen, legen Sie eine AWS-Ressource an. Daher hat jede Storage-Lens-Gruppe ihren eigenen Amazon-Ressourcennamen (ARN), den Sie angeben können, wenn Sie sie an ein S3-Storage-Lens-Dashboard anhängen oder daraus ausschließen.
> - Wenn eine Storage-Lens-Gruppe nicht an ein Dashboard angehängt ist, fallen für Sie keine zusätzlichen Gebühren für die Erstellung einer Storage-Lens-Gruppe an.
> - S3 Storage Lens aggregiert Nutzungsmetriken für ein Objekt aus allen passenden Storage-Lens-Gruppen. Wenn also ein Objekt den Filterbedingungen für zwei oder mehr Storage-Lens-Gruppen entspricht, werden Sie in Ihrer Speichernutzung wiederholte Angaben für dasselbe Objekt sehen.

Sie können eine Storage-Lens-Gruppe auf Kontoebene in einer bestimmten Heimatregion (aus der Liste der unterstützten AWS-Regionen) erstellen. Anschließend können Sie die Storage-Lens-Gruppe mehreren Storage-Lens-Dashboards zuordnen, sofern sich die Dashboards in demselben AWS-Konto und in derselben Heimatregion befinden. Sie sind in der Lage, pro Heimatregion in jedem AWS-Konto bis zu 50 Storage-Lens-Gruppen zu erstellen.

Die Verwaltung und Erstellung von S3-Storage-Lens-Gruppen ist mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface, (AWS CLI), der AWS-SDKs oder der Amazon-S3-REST-API möglich.

Themen

Anzeigen aggregierter Metriken von Storage-Lens-Gruppen

Sie können die aggregierten Metriken für Storage-Lens-Gruppen anzeigen, indem Sie die Gruppen an ein Dashboard anhängen. Die anzuhängenden Storage-Lens-Gruppen müssen sich in der angegebenen Heimatregion des Dashboard-Kontos befinden.

Um eine Storage-Lens-Gruppe an ein Dashboard anzuhängen, müssen Sie die Gruppe im Abschnitt Aggregation von Storage-Lens-Gruppen der Dashboard-Konfiguration angeben. Sind mehrere Storage-Lens-Gruppen vorhanden, können Sie die Ergebnisse der Aggregation von Storage-Lens-Gruppen filtern und bestimmte Gruppen ein- oder ausschließen. Weitere Informationen zum Anhängen von Gruppen an Dashboards finden Sie unter the section called “Anhängen oder Entfernen einer Storage-Lens-Gruppe”.

Nachdem Sie die Gruppen angehängt haben, erhalten Sie innerhalb von 48 Stunden die zusätzlichen Aggregationsdaten der Storage-Lens-Gruppen im Dashboard.

> **Note**
>
> Wenn Sie aggregierte Metriken für die Storage-Lens-Gruppe einsehen möchten, müssen Sie die Gruppe an das S3-Storage-Lens-Dashboard anhängen.

Berechtigungen für Storage-Lens-Gruppen

Storage-Lens-Gruppen erfordern gewisse Berechtigungen in AWS Identity and Access Management (IAM), um den Zugriff auf S3-Storage-Lens-Gruppen-Aktionen zu autorisieren. Zum Erteilen dieser Berechtigungen können Sie eine identitätsbasierte IAM-Richtlinie verwenden. Die Richtlinie lässt sich an IAM-Benutzer, -Gruppen oder -Rollen anhängen, wodurch diese die entsprechenden Berechtigungen erhalten. Solche Berechtigungen können die Fähigkeit beinhalten, Storage-Lens-Gruppen zu erstellen oder zu löschen, ihre Konfigurationen einzusehen oder ihre Tags zu verwalten.

Der IAM-Benutzer oder die IAM-Rolle, dem bzw. der Sie Berechtigungen gewähren, muss zu dem Konto gehören, das die Storage-Lens-Gruppe erstellt hat oder dem diese angehört.

Zur Nutzung von Storage-Lens-Gruppen und zur Anzeige der Metriken von Storage-Lens-Gruppen benötigen Sie zunächst die entsprechenden Berechtigungen für die Verwendung von S3 Storage Lens. Weitere Informationen finden Sie unter the section called “Berechtigungen für S3 Storage Lens”.

Für die Erstellung und Verwaltung von S3-Storage-Lens-Gruppen sind je nachdem, welche Aktionen Sie ausführen möchten, die folgenden IAM-Berechtigungen erforderlich:

Aktion	IAM-Berechtigungen
Erstellen einer neuen Storage-Lens-Gruppe	`s3:CreateStorageLensGroup`
Erstellen einer neuen Storage-Lens-Gruppe	`s3:CreateStorageLensGroup` , `s3:TagResource`
Aktualisieren einer vorhandenen Storage-Lens-Gruppe	`s3:UpdateStorageLensGroup`
Zurückgeben der Konfigurationsdetails einer Storage-Lens-Gruppe	`s3:GetStorageLensGroup`
Auflisten aller Storage-Lens-Gruppen in der Heimatregion	`s3:ListStorageLensGroups`
Löschen einer Storage-Lens-Gruppe	`s3:DeleteStorageLensGroup`
Auflisten der Tags, die der Storage-Lens-Gruppe hinzugefügt wurden	`s3:ListTagsForResource`

Aktion	IAM-Berechtigungen
Hinzufügen oder Aktualisieren eines Storage-Lens-Gruppen-Tags für eine vorhandene Storage-Lens-Gruppe	`s3:TagResource`
Löschen eines Tags aus einer Storage-Lens-Gruppe	`s3:UntagResource`

Im Folgenden finden Sie ein Beispiel für die Konfiguration der IAM-Richtlinie in dem Konto, in dem die Storage-Lens-Gruppe erstellt wurde. Zur Verwendung dieser Richtlinie ersetzen Sie *us-east-1* durch die Heimatregion, in der sich die Storage-Lens-Gruppe befindet. Ersetzen Sie *111122223333* durch Ihre AWS-Konto-ID und *example-storage-lens-group* durch den Namen der Storage-Lens-Gruppe. Wenn Sie diese Berechtigungen auf alle Storage-Lens-Gruppen anwenden möchten, ersetzen Sie *example-storage-lens-group* durch ein ***.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EXAMPLE-Statement-ID",
            "Effect": "Allow",
            "Action": [
                "s3:CreateStorageLensGroup",
                "s3:UpdateStorageLensGroup",
                "s3:GetStorageLensGroup",
                "s3:ListStorageLensGroups",
                "s3:DeleteStorageLensGroup,
                "s3:TagResource",
                "s3:UntagResource",
                "s3:ListTagsForResource"
                ],
            "Resource": "arn:aws:s3:us-east-1:111122223333:storage-lens-group/example-storage-lens-group"
        }
    ]
}
```

Weitere Informationen zur Verwendung von S3-Storage-Lens-Berechtigungen finden Sie unter Berechtigungen für Amazon S3 Storage Lens. Weitere Informationen zur IAM-Richtliniensprache finden Sie unter Richtlinien und Berechtigungen in Amazon S3.

Konfiguration von Storage-Lens-Gruppen

Namen von S3-Storage-Lens-Gruppen

Wir empfehlen, Storage-Lens-Gruppen Namen zu geben, die auf ihren Zweck hinweisen. Auf diese Weise können Sie leicht feststellen, an welche Gruppen Dashboards angehängt werden sollen. Um eine Storage-Lens-Gruppe an ein Dashboard anzuhängen, müssen Sie die Gruppe im Abschnitt Aggregation von Storage-Lens-Gruppen der Dashboard-Konfiguration angeben.

Namen von Storage-Lens-Gruppen müssen innerhalb des Kontos eindeutig sein. Sie dürfen nicht länger als 64 Zeichen sein. Nur Buchstaben (a–z, A–Z), Zahlen (0–9), Bindestriche (-) und Unterstriche (_) sind zulässig.

Heimatregion

Die Heimatregion ist die AWS-Region, in der eine Storage-Lens-Gruppe erstellt und verwaltet wird. Eine Storage-Lens-Gruppe wird in derselben Heimatregion wie das Amazon-S3-Storage-Lens-Dashboard erstellt. Die Konfiguration und die Metriken der Storage-Lens-Gruppe werden ebenfalls in dieser Region gespeichert. Sie sind in der Lage, in einer Heimatregion bis zu 50 Storage-Lens-Gruppen zu erstellen.

Nachdem Sie eine Storage-Lens-Gruppe erstellt haben, können Sie die Heimatregion nicht bearbeiten.

Scope

Um Objekte in eine Storage-Lens-Gruppe aufzunehmen, müssen sie sich im Umfang des Amazon-S3-Storage-Lens-Dashboards befinden. Der Umfang des Storage-Lens-Dashboards wird durch die Buckets bestimmt, die Sie in den Dashboard-Umfang der S3-Storage-Lens-Dashboard-Konfiguration aufgenommen haben.

Sie können verschiedene Filter für Ihre Objekte verwenden, um den Umfang einer Storage-Lens-Gruppe zu definieren. Um diese Storage-Lens-Gruppenmetriken im S3-Storage-Lens-Dashboard anzuzeigen, müssen Objekte den Filtern entsprechen, die Sie in die Storage-Lens-Gruppen aufnehmen. Angenommen, in einer Storage-Lens-Gruppe werden Objekte mit dem Präfix

`marketing` und dem Suffix `.png` berücksichtigt, aber keine Objekte entsprechen diesen Kriterien. In diesem Fall werden in dem täglichen Metrikexport keine Metriken für diese Storage-Lens-Gruppe generiert und im Dashboard sind keine Metriken für diese Gruppe sichtbar.

Filter

Sie können die folgenden Filter in einer S3-Storage-Lens-Gruppe verwenden:

- Präfixe – Gibt das Präfix der berücksichtigten Objekte an. Es handelt sich um eine Zeichenfolge am Anfang des Objektschlüsselnamens. Der Wert `images` für den Filter Präfixe schließt beispielsweise Objekte mit einem der folgenden Präfixe ein:`images/`, `images-marketing` und `images/production`. Die maximale Länge eines Präfixes beträgt 512 Byte.
- Suffixe – Gibt das Suffix der berücksichtigten Objekte an (z. B. `.png`, `.jpeg` oder `.csv`). Die maximale Länge eines Suffixes beträgt 1 024 Byte.
- Objekt-Tags – Gibt die Liste der Objekt-Tags an, nach denen gefiltert werden soll. Ein Tag-Schlüssel darf nicht mehr als 128 Unicode-Zeichen beinhalten, ein Tag-Wert maximal 256 Unicode-Zeichen. Beachten Sie, dass S3-Storage-Lens-Gruppen das Objekt nur anderen Objekten zuordnen, die auch leere Tag-Werte haben, wenn das Feld Objekt-Tag-Wert leer gelassen wird.
- Alter – Gibt den Altersbereich der enthaltenen Objekte in Tagen an. Es werden nur Ganzzahlen unterstützt.
- Größe – Gibt den Objektgrößenbereich der berücksichtigten Objekte in Byte an. Es werden nur Ganzzahlen unterstützt. Der maximal zulässige Wert ist 5 TB.

Objekt-Tags von Storage-Lens-Gruppen

Sie können eine Storage-Lens-Gruppe erstellen, die bis zu 10 Objekt-Tag-Filter enthält. Das folgende Beispiel enthält zwei Schlüssel-Wert-Paare für Objekt-Tags als Filter für eine Storage-Lens-Gruppe namens *`Marketing-Department`*. Zur Verwendung dieses Beispiels ersetzen Sie *`Marketing-Department`* durch den Namen Ihrer Gruppe und *`object-tag-key-1`*, *`object-tag-value-1`* usw. durch die Schlüssel-Wert-Paare der Objekt-Tags, nach denen Sie filtern möchten.

```
{
    "Name": "Marketing-Department",
    "Filter": {
     "MatchAnyTag":[
                {
                    "Key": "object-tag-key-1",
                    "Value": "object-tag-value-1"
```

```
                },
                {
                    "Key": "object-tag-key-2",
                    "Value": "object-tag-value-2"
                }
            ]
    }
}
```

Logische Operatoren (**`And`** oder **`Or`**)

Um mehrere Filterbedingungen in eine Storage-Lens-Gruppe aufzunehmen, können Sie logische Operatoren (entweder `And` oder `Or`) einsetzen. Im folgenden Beispiel besitzt die Storage-Lens-Gruppe *`Marketing-Department`* einen `And`-Operator, der die Filter `ObjectSize`, `Prefix` und `ObjectAge` enthält. Aufgrund des `And`-Operators werden nur Objekte, die allen Filterbedingungen entsprechen, in den Umfang der Storage-Lens-Gruppe aufgenommen.

Zur Verwendung dieses Beispiels ersetzen Sie die *`user input placeholders`* durch die Werte, nach denen Sie filtern möchten.

```
{
    "Name": "Marketing-Department",
    "Filter": {
        "And": {
            "MatchAnyPrefix": [
                "prefix-1",
                "prefix-2",
                "prefix-3/sub-prefix-1"
            ],
             "MatchObjectAge": {
                "DaysGreaterThan": 10,
                "DaysLessThan": 60
            },
            "MatchObjectSize": {
                "BytesGreaterThan": 10,
                "BytesLessThan": 60
            }
        }
    }
}
```

> **Note**
>
> Wenn Sie Objekte berücksichtigen möchten, die nur einer Filterbedingung entsprechen, ersetzen Sie in diesem Beispiel den logischen Operator `And` durch den logischen Operator `Or`.

AWS-Ressourcen-Tags

Jede S3-Storage-Lens-Gruppe gilt als AWS-Ressource mit eigenem Amazon-Ressourcenname (ARN). Daher können Sie bei der Konfiguration einer Storage-Lens-Gruppe optional AWS-Ressourcen-Tags zur Gruppe hinzufügen. Es ist möglich, für jede Storage-Lens-Gruppe bis zu 50 Tags hinzuzufügen. Zum Erstellen einer Storage-Lens-Gruppe mit Tags benötigen Sie die Berechtigungen `s3:TagResource` und `s3:CreateStorageLensGroup`.

Mithilfe von AWS-Ressourcen-Tags lassen sich Ressourcen nach Abteilung, Geschäftsbereich oder Projekt kategorisieren. Dies ist sinnvoll, wenn viele Ressourcen desselben Typs vorliegen. Durch die Anwendung von Tags können Sie eine bestimmte Storage-Lens-Gruppe auf Grundlage der ihr zugewiesenen Tags schnell identifizieren. Sie können Tags auch verwenden, um Ihre Kosten zu kategorisieren und zu verfolgen.

Wenn Sie einer Storage-Lens-Gruppe ein AWS-Ressourcen-Tag hinzufügen, aktivieren Sie außerdem die attributbasierte Zugriffskontrolle (ABAC). ABAC ist eine Autorisierungsstrategie, die Berechtigungen basierend auf Attributen – in diesem Fall Tags – definiert. Sie können auch Bedingungen verwenden, die Ressourcen-Tags in den IAM-Richtlinien für die Steuerung des Zugriffs auf AWS-Ressourcen angeben.

Sie können Tag (Markierung)-Schlüssel und -Werte bearbeiten und Tags (Markierungen) jederzeit von einer Ressource entfernen. Beachten Sie folgende Einschränkungen:

- Bei Tag-Schlüsseln und Tag-Werten muss die Groß- und Kleinschreibung beachtet werden.
- Wenn Sie ein Tag (Markierung) mit demselben Schlüssel wie ein vorhandener Tag (Markierung) für die Ressource hinzufügen, wird der alte Wert mit dem neuen überschrieben.
- Wenn Sie eine Ressource löschen, werden alle Tags (Markierungen) der Ressource ebenfalls gelöscht.
- Fügen Sie keine privaten oder vertraulichen Daten in AWS-Ressourcen-Tags ein.
- Systemtags (oder Tags mit Tag-Schlüsseln, die mit `aws:` beginnen) werden nicht unterstützt.

- Die einzelnen Tag-Schlüssel dürfen maximal 128 Zeichen lang sein. Die einzelnen Tag-Werte dürfen maximal 256 Zeichen lang sein.

Metrikexport für Storage-Lens-Gruppen

Die Metriken einer S3-Storage-Lens-Gruppe sind im Amazon-S3-Storage-Lens-Metrikexport für das Dashboard enthalten, an das die Storage-Lens-Gruppe angehängt wird. Allgemeine Informationen zur Exportfunktion für Storage-Lens-Metriken finden Sie unter Anzeigen von Amazon S3-Storage-Lens-Metriken mit einem Datenexport.

Der Metrikexport für Storage-Lens-Gruppen umfasst alle S3-Storage-Lens-Metriken, die im Umfang für das Dashboard enthalten sind, an das die Storage-Lens-Gruppe angehängt wurde. Der Export beinhaltet zudem zusätzliche Metrikdaten für Storage-Lens-Gruppen.

Nach Erstellung der Storage-Lens-Gruppe wird der Metrikexport täglich an den Bucket gesendet, den Sie bei der Konfiguration des Metrikexports für das Dashboard ausgewählt haben, an das die Gruppe angehängt wird. Es kann bis zu 48 Stunden dauern, bis Sie den ersten Metrikexport erhalten.

Damit Metriken im täglichen Export generiert werden, müssen Objekte den Filtern entsprechen, die Sie in die Storage-Lens-Gruppen aufnehmen. Wenn den Filtern, die Sie in die Storage-Lens-Gruppe eingebunden haben, keine Objekte entsprechen, werden keine Metriken generiert. Entspricht ein Objekt jedoch zwei oder mehr Storage-Lens-Gruppen, wird das Objekt im Metrikexport für jede Gruppe separat aufgeführt.

Sie können Metriken für Storage-Lens-Gruppen ermitteln, indem Sie in der Spalte `record_type` des Metrikexports für das Dashboard nach einem der folgenden Werte suchen:

- `STORAGE_LENS_GROUP_BUCKET`
- `STORAGE_LENS_GROUP_ACCOUNT`

In der Spalte `record_value` wird der Ressourcen-ARN für die Storage-Lens-Gruppe angezeigt (z. B. `arn:aws:s3:`*`us-east-1`*`:`*`111122223333`*`:storage-lens-group/`*`Marketing-Department`*).

Verwenden von Storage-Lens-Gruppen

Amazon-S3-Storage-Lens-Gruppen aggregieren Metriken mithilfe benutzerdefinierter Filter auf der Grundlage von Objektmetadaten. Sie können S3-Storage-Lens-Metriken anhand der Präfixe, der

Suffixe, der Objekt-Tags, der Objektgröße oder des Objektalters analysieren und filtern. Zudem bieten Amazon-S3-Storage-Lens-Gruppen die Möglichkeit, die Nutzung innerhalb und zwischen Amazon-S3-Buckets zu kategorisieren. Auf diese Weise erhalten Sie mehr Informationen zum S3-Speicher und können ihn besser optimieren.

Für die Visualisierung der Daten für eine Storage-Lens-Gruppe müssen Sie zunächst Ihre Storage-Lens-Gruppe an ein S3-Storage-Lens-Dashboard anhängen. Wenn Sie Storage-Lens-Gruppen im Dashboard verwalten müssen, können Sie die Dashboard-Konfiguration bearbeiten. Um festzustellen, welche Storage-Lens-Gruppen sich in Ihrem Konto befinden, können Sie sie auflisten. Auf der Registerkarte Storage-Lens-Gruppen im Dashboard können Sie jederzeit überprüfen, welche Storage-Lens-Gruppen an das Dashboard angefügt sind. Wenn Sie den Umfang einer vorhandenen Storage-Lens-Gruppe überprüfen oder aktualisieren möchten, können Sie deren Details einsehen. Storage-Lens-Gruppen können auch dauerhaft gelöscht werden.

Zum Verwalten von Berechtigungen erstellen Sie benutzerdefinierte AWS-Ressourcen-Tags und fügen sie den Storage-Lens-Gruppen hinzu. Mithilfe von AWS-Ressourcen-Tags lassen sich Ressourcen nach Abteilung, Geschäftsbereich oder Projekt kategorisieren. Dies ist sinnvoll, wenn viele Ressourcen desselben Typs vorliegen. Durch die Anwendung von Tags können Sie eine bestimmte Storage-Lens-Gruppe auf Grundlage der ihr zugewiesenen Tags schnell identifizieren.

Wenn Sie einer Storage-Lens-Gruppe ein AWS-Ressourcen-Tag hinzufügen, aktivieren Sie außerdem die attributbasierte Zugriffskontrolle (ABAC). ABAC ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen – in diesem Fall Tags – definiert werden. Sie können auch Bedingungen verwenden, die Ressourcen-Tags in den IAM-Richtlinien für die Steuerung des Zugriffs auf AWS-Ressourcen angeben.

Themen

Erstellen einer Storage-Lens-Gruppe

Die folgenden Beispiele zeigen, wie Sie eine Amazon-S3-Storage-Lens-Gruppe mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java erstellen.

Verwenden der S3-Konsole

So erstellen Sie eine Storage-Lens-Gruppe

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Wählen Sie Storage-Lens-Gruppe erstellen aus.
4. Geben Sie unter Allgemein den Namen der Storage-Lens-Gruppe und der Heimatregion ein.
5. Wählen Sie unter Umfang den Filter aus, der auf die Storage-Lens-Gruppe angewendet werden soll. Zum Anwenden mehrerer Filter wählen Sie die Filter und dann den logischen Operator UND oder ODER aus.

   - Wählen Sie für den Filter Präfixe die Option Präfixe aus und geben Sie eine Präfixzeichenfolge ein. Zum Hinzufügen mehrerer Präfixe wählen Sie Präfix hinzufügen aus. Zum Entfernen eines Präfixes wählen Sie neben dem zu entfernenden Präfix Entfernen aus.
   - Wählen Sie für den Filter Objekt-Tags die Option Objekt-Tags aus und geben Sie das Schlüssel-Wert-Paar für das Objekt ein. Wählen Sie dann Tag hinzufügen aus. Zum Entfernen eines vorhandenen Tags wählen Sie neben dem zu entfernenden Tag Entfernen aus.
   - Wählen Sie für den Filter Suffixe die Option Suffixe aus und geben Sie eine Suffixzeichenfolge ein. Um mehrere Suffixe hinzuzufügen, wählen Sie Suffix hinzufügen aus. Zum Entfernen eines Suffixes wählen Sie neben dem zu entfernenden Suffix Entfernen aus.
   - Geben Sie für den Filter Alter den Altersbereich des Objekts in Tagen an. Wählen Sie Mindestobjektalter festlegen aus und geben Sie das Mindestalter des Objekts ein. Wählen Sie dann Das maximale Objektalter festlegen aus und geben Sie das maximale Objektalter ein.
   - Geben Sie für den Filter Größe den Objektgrößenbereich und die Maßeinheit an. Wählen Sie Mindestobjektgröße festlegen aus und geben Sie die Mindestobjektgröße ein. Wählen Sie Die maximale Objektgröße festlegen aus und geben Sie die maximale Objektgröße ein.
6. (Optional) Fügen Sie für AWS-Ressourcen-Tags das Schlüssel-Wert-Paar hinzu und wählen Sie dann Tag hinzufügen aus.
7. Wählen Sie Storage-Lens-Gruppe erstellen aus.

Verwendung von AWS CLI

Mit dem folgenden AWS CLI-Beispielbefehl wird eine Storage-Lens-Gruppe erstellt. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control create-storage-lens-group --account-id 111122223333 \
--region us-east-1 --storage-lens-group=file://./marketing-department.json
```

Mit dem folgenden AWS CLI-Beispielbefehl wird eine Storage-Lens-Gruppe mit zwei AWS-Ressourcen-Tags erstellt. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control create-storage-lens-group --account-id 111122223333 \
--region us-east-1 --storage-lens-group=file://./marketing-department.json \
--tags Key=k1,Value=v1 Key=k2,Value=v2
```

JSON-Beispielkonfigurationen finden Sie unter Konfiguration von Storage-Lens-Gruppen.

Verwenden des AWS-SDKs für Java

In dem folgenden AWS SDK for Java-Beispiel wird eine Storage-Lens-Gruppe erstellt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

Example – Erstellen einer Storage-Lens-Gruppe mit einem einzigen Filter

In dem folgenden Beispiel wird eine Storage-Lens-Gruppe namens *Marketing-Department* erstellt. Diese Gruppe besitzt einen Filter für das Objektalter, der einen Altersbereich von *30* bis *90* Tagen festlegt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.MatchObjectAge;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
```

```
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;

public class CreateStorageLensGroupWithObjectAge {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            StorageLensGroupFilter objectAgeFilter = StorageLensGroupFilter.builder()
                    .matchObjectAge(MatchObjectAge.builder()
                            .daysGreaterThan(30)
                            .daysLessThan(90)
                            .build())
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(objectAgeFilter)
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
 CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example — Erstellen einer Storage-Lens-Gruppe mit einem **AND**-Operator, der mehrere Filter umfasst

In dem folgenden Beispiel wird eine Storage-Lens-Gruppe namens *Marketing-Department* erstellt. Diese Gruppe verwendet den AND-Operator, um anzugeben, dass Objekte allen Filterbedingungen entsprechen müssen. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.MatchObjectAge;
import software.amazon.awssdk.services.s3control.model.MatchObjectSize;
import software.amazon.awssdk.services.s3control.model.S3Tag;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupAndOperator;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;


public class CreateStorageLensGroupWithAndFilter {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            // Create object tags.
            S3Tag tag1 = S3Tag.builder()
                    .key("object-tag-key-1")
                    .value("object-tag-value-1")
                    .build();
            S3Tag tag2 = S3Tag.builder()
                    .key("object-tag-key-2")
                    .value("object-tag-value-2")
                    .build();

            StorageLensGroupAndOperator andOperator =
 StorageLensGroupAndOperator.builder()
                    .matchAnyPrefix("prefix-1", "prefix-2", "prefix-3/sub-prefix-1")
```

```
                    .matchAnySuffix(".png", ".gif", ".jpg")
                    .matchAnyTag(tag1, tag2)
                    .matchObjectAge(MatchObjectAge.builder()
                            .daysGreaterThan(30)
                            .daysLessThan(90).build())
                    .matchObjectSize(MatchObjectSize.builder()
                            .bytesGreaterThan(1000L)
                            .bytesLessThan(6000L).build())
                    .build();

            StorageLensGroupFilter andFilter = StorageLensGroupFilter.builder()
                    .and(andOperator)
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(andFilter)
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
 CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example — Erstellen einer Storage-Lens-Gruppe mit einem **OR**-Operator, der mehrere Filter umfasst

In dem folgenden Beispiel wird eine Storage-Lens-Gruppe namens *Marketing-Department* erstellt. Für diese Gruppe wird ein OR-Operator verwendet, um einen Präfixfilter (*prefix-1*, *prefix-2*, *prefix3/sub-prefix-1*) oder einen Objektgrößenfilter mit einem Größenbereich zwischen *1000* Byte und *6000* Byte anzuwenden. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.MatchObjectSize;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupOrOperator;

public class CreateStorageLensGroupWithOrFilter {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            StorageLensGroupOrOperator orOperator =
 StorageLensGroupOrOperator.builder()
                    .matchAnyPrefix("prefix-1", "prefix-2", "prefix-3/sub-prefix-1")
                    .matchObjectSize(MatchObjectSize.builder()
                            .bytesGreaterThan(1000L)
                            .bytesLessThan(6000L)
                            .build())
                    .build();

            StorageLensGroupFilter orFilter = StorageLensGroupFilter.builder()
                    .or(orOperator)
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(orFilter)
```

```
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
 CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example – Erstellen einer Storage-Lens-Gruppe mit einem einzigen Filter und zwei AWS-Ressourcen-Tags

In dem folgenden Beispiel wird eine Storage-Lens-Gruppe namens *Marketing-Department* erstellt, die einen Suffixfilter aufweist. In diesem Beispiel werden der Storage-Lens-Gruppe außerdem zwei AWS-Ressourcen-Tags hinzugefügt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;
import software.amazon.awssdk.services.s3control.model.Tag;
```

```
public class CreateStorageLensGroupWithResourceTags {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            // Create AWS resource tags.
            Tag resourceTag1 = Tag.builder()
                    .key("resource-tag-key-1")
                    .value("resource-tag-value-1")
                    .build();
            Tag resourceTag2 = Tag.builder()
                    .key("resource-tag-key-2")
                    .value("resource-tag-value-2")
                    .build();

            StorageLensGroupFilter suffixFilter = StorageLensGroupFilter.builder()
                    .matchAnySuffix(".png", ".gif", ".jpg")
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(suffixFilter)
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
 CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .tags(resourceTag1, resourceTag2)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
```

```
            e.printStackTrace();
        }
    }
}
```

JSON-Beispielkonfigurationen finden Sie unter Konfiguration von Storage-Lens-Gruppen.

Anhängen von S3-Storage-Lens-Gruppen an das Dashboard und Entfernen der Gruppen

Nachdem Sie in Amazon S3 Storage Lens ein Upgrade auf das erweiterte Kontingent durchgeführt haben, können Sie eine Storage-Lens-Gruppe an das Dashboard anhängen. Sind mehrere Storage-Lens-Gruppen vorhanden, lassen sich die gewünschten Gruppen ein- oder ausschließen.

Die Storage-Lens-Gruppen müssen sich in der angegebenen Heimatregion im Dashboard-Konto befinden. Nachdem Sie eine Storage-Lens-Gruppe an das Dashboard angehängt haben, erhalten Sie innerhalb von 48 Stunden die Aggregationsdaten der zusätzlichen Storage-Lens-Gruppe im Metrikexport.

> **Note**
>
> Wenn Sie aggregierte Metriken für die Storage Lens-Gruppe einsehen möchten, müssen Sie sie an das Storage-Lens-Dashboard anhängen. Beispiele für JSON-Konfigurationsdateien für eine Storage Lens-Gruppe finden Sie unter Beispiel für S3-Storage-Lens-Konfiguration mit Storage-Lens-Gruppen in JSON.

Anhängen einer Storage-Lens-Gruppe an ein S3-Storage-Lens-Dashboard

So hängen Sie eine Storage-Lens-Gruppe an ein Storage-Lens-Dashboard an

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich unter Storage Lens die Option Dashboards aus.
3. Aktivieren Sie das Optionsfeld für das Storage-Lens-Dashboard, an das Sie eine Storage-Lens-Gruppe anhängen möchten.
4. Wählen Sie Edit (Bearbeiten) aus.
5. Unter Metrikauswahl, wählen Sie Fortschrittliche Metriken und Empfehlungen aus.
6. Wählen Sie Aggregation von Storage-Lens-Gruppen aus.

> **Note**
>
> Standardmäßig ist die Option Erweiterte Metriken auch ausgewählt. Sie können diese Einstellung jedoch auch deaktivieren, da sie nicht erforderlich ist, um Daten von Storage-Lens-Gruppen zu aggregieren.

7. Scrollen Sie nach unten zu Aggregation von Storage-Lens-Gruppen und geben Sie die Storage-Lens-Gruppe(n) an, die Sie in der Datenaggregation berücksichtigen oder daraus ausschließen möchten. Sie können die folgenden Filteroptionen verwenden:

   - Wenn Sie bestimmte Storage-Lens-Gruppen berücksichtigen möchten, wählen Sie Storage-Lens-Gruppen einbeziehen aus. Wählen Sie unter Zu berücksichtigende Storage-Lens-Gruppen Ihre Storage-Lens-Gruppen aus.
   - Wenn Sie alle Storage-Lens-Gruppen berücksichtigen möchten, wählen Sie Alle Storage-Lens-Gruppen in der Heimatregion in diesem Konto berücksichtigen aus.
   - Möchten Sie bestimmte Storage-Lens-Gruppen ausschließen, wählen Sie Storage-Lens-Gruppen ausschließen aus. Wählen Sie unter Auszuschließende Storage-Lens-Gruppen die Storage-Lens-Gruppen aus, die Sie ausschließen möchten.

8. Wählen Sie Save Changes. Nachdem Sie eine Storage-Lens-Gruppe an das Dashboard angehängt haben, erhalten Sie innerhalb von 48 Stunden die Aggregationsdaten der zusätzlichen Storage-Lens-Gruppe im Metrikexport.

Entfernen einer Storage-Lens-Gruppe aus einem S3-Storage-Lens-Dashboard

So entfernen Sie eine Storage-Lens-Gruppe aus einem S3-Storage-Lens-Dashboard

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich unter Storage Lens die Option Dashboards aus.
3. Aktivieren Sie das Optionsfeld für das Storage-Lens-Dashboard, aus dem Sie eine Storage-Lens-Gruppe entfernen möchten.
4. Wählen Sie Dashboard-Konfigurationen anzeigen aus.
5. Wählen Sie Edit (Bearbeiten) aus.
6. Scrollen Sie nach unten bis zum Abschnitt Auswahl von Metriken.

7. Wählen Sie unter Aggregation von Storage-Lens-Gruppen das X neben der Storage-Lens-Gruppe aus, die Sie entfernen möchten. Dadurch wird die Storage-Lens-Gruppe entfernt.

   Wenn Sie alle Storage-Lens-Gruppen in das Dashboard aufgenommen haben, deaktivieren Sie das Kontrollkästchen neben Alle Storage-Lens-Gruppen in der Heimatregion in diesem Konto berücksichtigen.

8. Wählen Sie Save Changes.

> **Note**
>
> Es dauert bis zu 48 Stunden, bis die Konfigurationsänderungen im Dashboard angezeigt werden.

Visualisieren von Storage Lens-Gruppendaten

Sie können Storage-Lens-Gruppendaten visualisieren, indem Sie die Gruppe an das Amazon-S3-Storage-Lens-Dashboard anhängen. Nachdem Sie die Storage-Lens-Gruppe in die Aggregation von Storage-Lens-Gruppen in der Dashboard-Konfiguration aufgenommen haben, kann es bis zu 48 Stunden dauern, bis die Daten der Storage-Lens-Gruppe im Dashboard angezeigt werden.

Nach Aktualisierung der Dashboard-Konfiguration werden alle neu hinzugefügten Storage-Lens-Gruppen in der Liste der verfügbaren Ressourcen auf der Registerkarte Storage-Lens-Gruppen angezeigt. Zudem können Sie die Speichernutzung auf der Registerkarte Übersicht weiter analysieren, indem Sie die Daten nach einer anderen Dimension aufteilen. Wählen Sie z. B. eines der Elemente aus, die in den drei wichtigsten Kategorien aufgeführt sind, und dann Analysieren nach, um die Daten nach einer anderen Dimension zu aufzuteilen. Sie können nicht dieselbe Dimension wie der Filter selbst anwenden.

> **Note**
>
> Es ist nicht möglich, einen Storage-Lens-Gruppenfilter zusammen mit einem Präfixfilter anzuwenden oder umgekehrt. Sie können eine Storage-Lens-Gruppe auch nicht weiter analysieren, indem Sie einen Präfixfilter verwenden.

Mithilfe der Registerkarte Storage-Lens-Gruppen im Amazon-S3-Storage-Lens-Dashboard lässt sich die Datenvisualisierung für die Storage-Lens-Gruppen anzupassen, die mit dem Dashboard verknüpft

sind. Sie können entweder die Daten für einige Storage-Lens-Gruppen, die an das Dashboard angehängt sind, oder für alle Gruppen visualisieren.

Beachten Sie bei der Visualisierung von Storage-Lens-Gruppendaten im S3-Storage-Lens-Dashboard Folgendes:

- S3 Storage Lens aggregiert Nutzungsmetriken für ein Objekt aus allen passenden Storage-Lens-Gruppen. Wenn also ein Objekt den Filterbedingungen für zwei oder mehr Storage-Lens-Gruppen entspricht, werden Sie in Ihrer Speichernutzung wiederholte Zählungen für dasselbe Objekt sehen.
- Objekte müssen den Filtern entsprechen, die Sie in die Storage-Lens-Gruppen einbinden. Wenn den Filtern, die Sie in die Storage-Lens-Gruppe eingebunden haben, keine Objekte entsprechen, werden keine Metriken generiert. Um festzustellen, ob es nicht zugewiesene Objekte gibt, überprüfen Sie die Gesamtzahl der Objekte im Dashboard auf Konto- und Bucket-Ebene.

Aktualisieren einer Storage-Lens-Gruppe

In den folgenden Beispielen wird veranschaulicht, wie eine Amazon-S3-Storage-Lens-Gruppe aktualisiert wird. Sie können eine Storage-Lens-Gruppe mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java aktualisieren.

Verwenden der S3-Konsole

So aktualisieren Sie eine Storage-Lens-Gruppe

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Wählen Sie unter Storage-Lens-Gruppen die zu aktualisierende Storage-Lens-Gruppe aus.
4. Wählen Sie unter Umfang die Option Bearbeiten aus.
5. Wählen Sie auf der Seite Umfang den Filter aus, der auf die Storage-Lens-Gruppe angewendet werden soll. Zum Anwenden mehrerer Filter wählen Sie die Filter und dann den logischen Operator UND oder ODER aus.

    - Wählen Sie für den Filter Präfixe die Option Präfixe aus und geben Sie eine Präfixzeichenfolge ein. Zum Hinzufügen mehrerer Präfixe wählen Sie Präfix hinzufügen aus. Zum Entfernen eines Präfixes wählen Sie neben dem zu entfernenden Präfix Entfernen aus.

- Geben Sie für den Filter Objekt-Tags das Schlüssel-Wert-Paar für das Objekt ein. Wählen Sie dann Tag hinzufügen aus. Zum Entfernen eines vorhandenen Tags wählen Sie neben dem zu entfernenden Tag Entfernen aus.
- Wählen Sie für den Filter Suffixe die Option Suffixe aus und geben Sie eine Suffixzeichenfolge ein. Um mehrere Suffixe hinzuzufügen, wählen Sie Suffix hinzufügen aus. Zum Entfernen eines Suffixes wählen Sie neben dem zu entfernenden Suffix Entfernen aus.
- Geben Sie für den Filter Alter den Altersbereich des Objekts in Tagen an. Wählen Sie Mindestobjektalter festlegen aus und geben Sie das Mindestalter des Objekts ein. Geben Sie für Das maximale Objektalter festlegen das maximale Objektalter ein.
- Geben Sie für den Filter Größe den Objektgrößenbereich und die Maßeinheit an. Wählen Sie Mindestobjektgröße festlegen aus und geben Sie die Mindestobjektgröße ein. Geben Sie für Die maximale Objektgröße festlegen die maximale Objektgröße ein.

6. Wählen Sie Save Changes. Die Detailseite für die Storage-Lens-Gruppe wird angezeigt.
7. (Optional) Wenn Sie ein neues AWS-Ressourcen-Tag hinzufügen möchten, scrollen Sie zum Abschnitt AWS-Ressourcen-Tags und wählen Sie dann Tags hinzufügen aus. Die Seite Add tags (Tags hinzufügen) wird angezeigt.

   Fügen Sie das neue Schlüssel-Wert-Paar hinzu und wählen Sie dann Änderungen speichern aus. Die Detailseite für die Storage-Lens-Gruppe wird angezeigt.
8. (Optional) Wenn Sie ein vorhandenes AWS-Ressourcen-Tag entfernen möchten, scrollen Sie zum Abschnitt AWS-Ressourcen-Tags und wählen Sie das Ressourcen-Tag aus. Wählen Sie dann Löschen. Das Dialogfeld AWS-Ressourcen-Tags löschen wird angezeigt.

   Wählen Sie erneut Löschen aus, um das AWS-Ressourcen-Tag dauerhaft zu löschen.

   > **Note**
   >
   > Nachdem Sie ein AWS-Ressourcen-Tag dauerhaft gelöscht haben, kann es nicht wiederhergestellt werden.

Verwendung von AWS CLI

Mit dem folgenden AWS CLI-Beispielbefehl werden die Konfigurationsdetails für eine Storage-Lens-Gruppe namens *`marketing-department`* zurückgegeben. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control get-storage-lens-group --account-id 111122223333 \
--region us-east-1 --name marketing-department
```

Im folgenden AWS CLI-Beispiel wird eine Storage-Lens-Gruppe aktualisiert. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control update-storage-lens-group --account-id 111122223333 \
--region us-east-1 --storage-lens-group=file://./marketing-department.json
```

JSON-Beispielkonfigurationen finden Sie unter Konfiguration von Storage-Lens-Gruppen.

Verwenden des AWS-SDKs für Java

Im folgenden AWS SDK for Java-Beispiel werden die Konfigurationsdetails für die Storage-Lens-Gruppe *Marketing-Department* im Konto *111122223333* zurückgegeben. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupResponse;

public class GetStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            GetStorageLensGroupRequest getRequest =
 GetStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .accountId(accountId).build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
```

```
                    .build();
            GetStorageLensGroupResponse response =
 s3ControlClient.getStorageLensGroup(getRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

In dem folgenden Beispiel wird die Storage-Lens-Gruppe *Marketing-Department* im Konto *111122223333* aktualisiert. In diesem Beispiel wird der Dashboard-Umfang so aktualisiert, dass die Objekte berücksichtigt werden, die einem der folgenden Suffixe entsprechen: *.png*, *.gif*, *.jpg* oder *.jpeg*. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;
import software.amazon.awssdk.services.s3control.model.UpdateStorageLensGroupRequest;

public class UpdateStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            // Create updated filter.
            StorageLensGroupFilter suffixFilter = StorageLensGroupFilter.builder()
                    .matchAnySuffix(".png", ".gif", ".jpg", ".jpeg")
                    .build();
```

```
            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(suffixFilter)
                    .build();

            UpdateStorageLensGroupRequest updateStorageLensGroupRequest =
 UpdateStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId)
                    .build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.updateStorageLensGroup(updateStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

JSON-Beispielkonfigurationen finden Sie unter Konfiguration von Storage-Lens-Gruppen.

Verwalten von AWS-Ressourcen-Tags mit Storage-Lens-Gruppen

Jede Amazon-S3-Storage-Lens-Gruppe gilt als AWS-Ressource mit eigenem Amazon-Ressourcennamen (ARN). Daher können Sie bei der Konfiguration einer Storage-Lens-Gruppe optional AWS-Ressourcen-Tags zur Gruppe hinzufügen. Es ist möglich, für jede Storage-Lens-Gruppe bis zu 50 Tags hinzuzufügen. Zum Erstellen einer Storage-Lens-Gruppe mit Tags benötigen Sie die Berechtigungen `s3:TagResource` und `s3:CreateStorageLensGroup`.

Mithilfe von AWS-Ressourcen-Tags lassen sich Ressourcen nach Abteilung, Geschäftsbereich oder Projekt kategorisieren. Dies ist sinnvoll, wenn viele Ressourcen desselben Typs vorliegen. Durch die Anwendung von Tags können Sie eine bestimmte Storage-Lens-Gruppe auf Grundlage der ihr

zugewiesenen Tags schnell identifizieren. Darüber hinaus sind Sie in der Lage, Kosten mithilfe von Tags zu verfolgen und zu verteilen.

Wenn Sie einer Storage-Lens-Gruppe ein AWS-Ressourcen-Tag hinzufügen, aktivieren Sie außerdem die attributbasierte Zugriffskontrolle (ABAC). ABAC ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen – in diesem Fall Tags – definiert werden. Sie können auch Bedingungen verwenden, die Ressourcen-Tags in den IAM-Richtlinien für die Steuerung des Zugriffs auf AWS-Ressourcen angeben.

Sie können Tag (Markierung)-Schlüssel und -Werte bearbeiten und Tags (Markierungen) jederzeit von einer Ressource entfernen. Beachten Sie die folgenden Einschränkungen:

- Bei Tag-Schlüsseln und -Werten wird zwischen Groß- und Kleinschreibung unterschieden.
- Wenn Sie ein Tag (Markierung) mit demselben Schlüssel wie ein vorhandener Tag (Markierung) für die Ressource hinzufügen, wird der alte Wert mit dem neuen überschrieben.
- Wenn Sie eine Ressource löschen, werden alle Tags (Markierungen) der Ressource ebenfalls gelöscht.
- Fügen Sie keine privaten oder vertraulichen Daten in AWS-Ressourcen-Tags ein.
- System-Tags (mit Tag-Schlüsseln, die mit `aws:` beginnen) werden nicht unterstützt.
- Die einzelnen Tag-Schlüssel dürfen maximal 128 Zeichen lang sein. Die einzelnen Tag-Werte dürfen maximal 256 Zeichen lang sein.

In den folgenden Beispielen wird veranschaulicht, wie AWS-Ressourcen-Tags mit Storage-Lens-Gruppen verwendet werden.

Themen

Hinzufügen eines AWS-Ressourcen-Tags zu einer Storage-Lens-Gruppe

In den folgenden Beispielen wird veranschaulicht, wie AWS-Ressourcen-Tags einer Amazon-S3-Storage-Lens-Gruppe hinzugefügt werden. Sie können Ressourcen-Tags mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java hinzufügen.

Verwenden der S3-Konsole

So fügen Sie ein AWS-Ressourcen-Tag zu einer Storage-Lens-Gruppe hinzu

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Wählen Sie unter Storage-Lens-Gruppen die zu aktualisierende Storage-Lens-Gruppe aus.
4. Wählen Sie unter AWS-Ressourcen-Tags die Option Tags hinzufügen aus.
5. Fügen Sie auf der Seite Tags hinzufügen das neue Schlüssel-Wert-Paar hinzu.

> **Note**
>
> Wenn Sie ein neues Tag hinzufügen, dessen Schlüssel mit dem eines vorhandenen Tags identisch ist, wird der vorherige Tag-Wert überschrieben.

6. (Optional) Um mehrere neue Tags hinzuzufügen, wählen Sie erneut Tag hinzufügen aus und fügen Sie weiterhin neue Einträge hinzu. Sie können der Storage-Lens-Gruppe bis zu 50 AWS-Ressourcen-Tags hinzufügen.
7. (Optional) Wenn Sie einen neu hinzugefügten Eintrag entfernen möchten, wählen Sie neben dem zu entfernenden Tag Entfernen aus.
8. Wählen Sie Save Changes.

Verwendung von AWS CLI

Mit dem folgenden AWS CLI-Beispielbefehl werden einer vorhandenen Storage-Lens-Gruppe namens *marketing-department* zwei Ressourcen-Tags hinzugefügt. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control tag-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/marketing-
department \
--region us-east-1 --tags Key=k1,Value=v1 Key=k2,Value=v2
```

Verwenden des AWS-SDKs für Java

Im folgenden AWS SDK for Java-Beispiel werden einer vorhandenen Storage-Lens-Gruppe zwei AWS-Ressourcen-Tags hinzugefügt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.Tag;
import software.amazon.awssdk.services.s3control.model.TagResourceRequest;

public class TagResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
        String accountId = "111122223333";

        try {
            Tag resourceTag1 = Tag.builder()
                .key("resource-tag-key-1")
                .value("resource-tag-value-1")
                .build();
            Tag resourceTag2 = Tag.builder()
                    .key("resource-tag-key-2")
                    .value("resource-tag-value-2")
                    .build();
            TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
                    .resourceArn(resourceARN)
                    .tags(resourceTag1, resourceTag2)
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.tagResource(tagResourceRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
```

```
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Aktualisieren von Tag-Werten einer Storage-Lens-Gruppe

In den folgenden Beispielen wird veranschaulicht, wie Sie die Tag-Werte für eine Storage-Lens-Gruppe mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java aktualisieren.

Verwenden der S3-Konsole

So aktualisieren Sie AWS-Ressourcen-Tags für eine Storage-Lens-Gruppe

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Wählen Sie unter Storage-Lens-Gruppen die zu aktualisierende Storage-Lens-Gruppe aus.
4. Wählen Sie unter AWS-Ressourcen-Tags das zu aktualisierende Tag aus.
5. Fügen Sie den neuen Tag-Wert hinzu. Verwenden Sie hierbei den Schlüssel, der dem Schlüssel des zu aktualisierenden Schlüssel-Wert-Paares entspricht. Klicken Sie auf das Häkchen-Symbol, um den Tag-Wert zu aktualisieren.

   > **Note**
   >
   > Wenn Sie ein neues Tag hinzufügen, dessen Schlüssel mit dem eines vorhandenen Tags identisch ist, wird der vorherige Tag-Wert überschrieben.

6. (Optional) Wenn Sie neue Tags hinzufügen möchten, wählen Sie Tag hinzufügen aus, um neue Einträge hinzuzufügen. Die Seite Add tags (Tags hinzufügen) wird angezeigt.

   Es ist möglich, für jede Storage-Lens-Gruppe bis zu 50 AWS-Ressourcen-Tags hinzuzufügen. Wenn Sie das Hinzufügen der neuen Tags beendet haben, klicken Sie auf Änderungen speichern.

7. (Optional) Wenn Sie einen neu hinzugefügten Eintrag entfernen möchten, wählen Sie neben dem zu entfernenden Tag Entfernen aus. Wenn Sie mit dem Entfernen der Tags fertig sind, wählen Sie Speichern aus.

Verwendung von AWS CLI

Mit dem folgenden AWS CLI-Beispielbefehl werden zwei Tag-Werte für die Storage-Lens-Gruppe namens *marketing-department* aktualisiert. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control tag-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/marketing-
department \
--region us-east-1 --tags Key=k1,Value=v3 Key=k2,Value=v4
```

Verwenden des AWS-SDKs für Java

In dem folgenden AWS SDK for Java-Beispiel werden zwei Storage-Lens-Gruppen-Tag-Werte aktualisiert. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.Tag;
import software.amazon.awssdk.services.s3control.model.TagResourceRequest;

public class UpdateTagsForResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
        String accountId = "111122223333";

        try {
            Tag updatedResourceTag1 = Tag.builder()
                .key("resource-tag-key-1")
                .value("resource-tag-updated-value-1")
                .build();
```

```
            Tag updatedResourceTag2 = Tag.builder()
                    .key("resource-tag-key-2")
                    .value("resource-tag-updated-value-2")
                    .build();
            TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
                    .resourceArn(resourceARN)
                    .tags(updatedResourceTag1, updatedResourceTag2)
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.tagResource(tagResourceRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Löschen eines AWS-Ressourcen-Tags aus einer Storage-Lens-Gruppe

In den folgenden Beispielen wird veranschaulicht, wie ein AWS-Ressourcen-Tag aus einer Storage-Lens-Gruppe gelöscht wird. Sie können Tags mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java löschen.

Verwenden der S3-Konsole

So löschen Sie ein AWS-Ressourcen-Tag aus einer Storage-Lens-Gruppe

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Wählen Sie unter Storage-Lens-Gruppen die zu aktualisierende Storage-Lens-Gruppe aus.
4. Wählen Sie unter AWS-Ressourcen-Tags das zu löschende Schlüssel-Wert-Paar aus.

5. Wählen Sie Delete (Löschen). Das Dialogfeld AWS-Ressourcen-Tags löschen wird angezeigt.

> **Note**
>
> Wenn Tags für die Zugriffskontrolle verwendet werden, kann die Durchführung dieser Aktion Auswirkungen auf zugehörige Ressourcen haben. Nachdem Sie ein Tag dauerhaft gelöscht haben, kann es nicht wiederhergestellt werden.

6. Wählen Sie Löschen aus, um das Schlüssel-Wert-Paar dauerhaft zu löschen.

Verwendung von AWS CLI

Mit dem folgenden AWS CLI-Befehl werden zwei AWS-Ressourcen-Tags aus einer vorhandenen Storage-Lens-Gruppe gelöscht. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie die *user input placeholders* durch Ihre Informationen.

```
aws s3control untag-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/Marketing-
Department \
--region us-east-1 --tag-keys k1 k2
```

Verwenden des AWS-SDKs für Java

Im folgenden AWS SDK for Java-Beispiel werden zwei AWS-Ressourcen-Tags aus der Storage-Lens-Gruppe mit dem Amazon-Ressourcennamen (ARN) gelöscht, den Sie im Konto *111122223333* angeben. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.UntagResourceRequest;

public class UntagResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
```

```
        String accountId = "111122223333";

        try {
            String tagKey1 = "resource-tag-key-1";
            String tagKey2 = "resource-tag-key-2";
            UntagResourceRequest untagResourceRequest = UntagResourceRequest.builder()
                    .resourceArn(resourceARN)
                    .tagKeys(tagKey1, tagKey2)
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.untagResource(untagResourceRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Auflisten von Storage-Lens-Gruppen-Tags

In den folgenden Beispielen wird veranschaulicht, wie die einer Storage-Lens-Gruppe zugeordneten AWS-Ressourcen-Tags aufgelistet werden. Sie können Tags mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java auflisten.

Verwenden der S3-Konsole

So überprüfen Sie die Liste der Tags und Tag-Werte für eine Storage-Lens-Gruppe

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Wählen Sie unter Storage-Lens-Gruppen die betreffende Storage-Lens-Gruppe aus.

4. Scrollen Sie nach unten zum Abschnitt AWS-Ressourcen-Tags. Alle benutzerdefinierten AWS-Ressourcen-Tags, die der Storage-Lens-Gruppe hinzugefügt wurden, sind zusammen mit ihren Tag-Werten aufgeführt.

Verwendung von AWS CLI

Mit dem folgenden AWS CLI-Beispielbefehl werden alle Tag-Werte für die Storage-Lens-Gruppe namens *marketing-department* aufgelistet. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control list-tags-for-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/marketing-department \
--region us-east-1
```

Verwenden des AWS-SDKs für Java

Im folgenden AWS SDK for Java-Beispiel werden die Storage-Lens-Gruppen-Tag-Werte für den angegebenen Amazon-Ressourcennamen (ARN) der Storage-Lens-Gruppe aufgelistet. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListTagsForResourceRequest;
import software.amazon.awssdk.services.s3control.model.ListTagsForResourceResponse;

public class ListTagsForResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
        String accountId = "111122223333";

        try {
            ListTagsForResourceRequest listTagsForResourceRequest =
 ListTagsForResourceRequest.builder()
                    .resourceArn(resourceARN)
```

```
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            ListTagsForResourceResponse response =
 s3ControlClient.listTagsForResource(listTagsForResourceRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Auflisten aller Storage-Lens-Gruppen

In den folgenden Beispielen wird veranschaulicht, wie alle Amazon-S3-Storage-Lens-Gruppen in einem AWS-Konto und in einer Heimatregion aufgelistet werden. Diese Beispiele zeigen, wie alle Storage-Lens-Gruppen mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java aufgelistet werden.

Verwenden der S3-Konsole

So listen Sie alle Storage-Lens-Gruppen in einem Konto und einer Heimatregion auf

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Unter Storage-Lens-Gruppen wird die Liste der Storage-Lens-Gruppen in Ihrem Konto angezeigt.

Verwendung von AWS CLI

Im folgenden AWS CLI-Beispiel werden alle Storage-Lens-Gruppen für Ihr Konto aufgelistet. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control list-storage-lens-groups --account-id 111122223333 \
--region us-east-1
```

Verwenden des AWS-SDKs für Java

Im folgenden Beispiel für AWS SDK for Java werden alle Storage-Lens-Gruppen für das Konto *111122223333* aufgelistet. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListStorageLensGroupsRequest;
import software.amazon.awssdk.services.s3control.model.ListStorageLensGroupsResponse;

public class ListStorageLensGroups {
    public static void main(String[] args) {
        String accountId = "111122223333";

        try {
            ListStorageLensGroupsRequest listStorageLensGroupsRequest =
 ListStorageLensGroupsRequest.builder()
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            ListStorageLensGroupsResponse response =
 s3ControlClient.listStorageLensGroups(listStorageLensGroupsRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
```

```
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Anzeigen von Details zu Storage-Lens-Gruppen

In den folgenden Beispielen wird veranschaulicht, wie die Konfigurationsdetails einer Amazon-S3-Storage-Lens-Gruppe angezeigt werden. Sie können diese Details mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java anzeigen.

Verwenden der S3-Konsole

So zeigen Sie die Konfigurationsdetails einer Storage-Lens-Gruppe an

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Wählen Sie unter Storage-Lens-Gruppen die Optionsschaltfläche neben der betreffenden Storage-Lens-Gruppe aus.
4. Wählen Sie die Option View details aus. Jetzt können Sie die Details der Storage-Lens-Gruppe überprüfen.

Verwendung von AWS CLI

Im folgenden AWS CLI-Beispiel werden die Konfigurationsdetails für eine Storage-Lens-Gruppe zurückgegeben. Wenn Sie dieses Beispielbefehl verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control get-storage-lens-group --account-id 111122223333 \
--region us-east-1 --name marketing-department
```

Verwenden des AWS-SDKs für Java

Im folgenden AWS SDK for Java-Beispiel werden die Konfigurationsdetails für die Storage-Lens-Gruppe namens *Marketing-Department* im Konto *111122223333* zurückgegeben. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupResponse;

public class GetStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            GetStorageLensGroupRequest getRequest =
 GetStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .accountId(accountId).build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            GetStorageLensGroupResponse response =
 s3ControlClient.getStorageLensGroup(getRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
```

```
    }
}
```

Löschen einer Storage-Lens-Gruppe

In den folgenden Beispielen wird veranschaulicht, wie Sie eine Amazon-S3-Storage-Lens-Gruppe mithilfe der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und des AWS SDK for Java löschen.

Verwenden der S3-Konsole

So löschen Sie eine Storage-Lens-Gruppe

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich die Option Storage-Lens-Gruppen aus.
3. Aktivieren Sie unter Storage-Lens-Gruppen das Optionsfeld neben der zu löschenden Storage-Lens-Gruppe.
4. Wählen Sie Delete (Löschen). Das Dialogfeld Storage-Lens-Gruppe löschen wird angezeigt.
5. Wählen Sie erneut Löschen aus, um die Storage-Lens-Gruppe dauerhaft zu löschen.

> **Note**
>
> Nachdem Sie eine Storage-Lens-Gruppe gelöscht haben, kann sie nicht wiederhergestellt werden.

Verwendung von AWS CLI

In dem folgenden AWS CLI-Beispiel wird die Storage-Lens-Gruppe *marketing-department* gelöscht. Wenn Sie diesen Beispielbefehl verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control delete-storage-lens-group --account-id 111122223333 \
--region us-east-1 --name marketing-department
```

Verwenden des AWS-SDKs für Java

In dem folgenden AWS SDK for Java-Beispiel wird die Storage-Lens-Gruppe *Marketing-Department* im Konto *111122223333* gelöscht. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.DeleteStorageLensGroupRequest;

public class DeleteStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            DeleteStorageLensGroupRequest deleteStorageLensGroupRequest =
 DeleteStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .accountId(accountId).build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.deleteStorageLensGroup(deleteStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Nachverfolgen von Amazon-S3-Anforderungen mit AWS X-Ray

AWS X-Ray erfasst Daten zu Anforderungen, die Ihre Anwendung verarbeitet. Sie können die Daten dann anzeigen und filtern, um Leistungsprobleme und Fehler in Ihrer verteilten Anwendungs- und Microservice-Architektur zu identifizieren und zu beheben. Für jede nachverfolgte Anforderung für Ihre Anwendung sehen Sie detaillierte Informationen zur Anforderung, zur Antwort und zu den Aufrufen, die Ihre Anwendung an nachgelagerte AWS-Ressourcen, Microservices, Datenbanken und HTTP-Web-APIs sendet.

Weitere Informationen finden Sie unter Was ist AWS X-Ray? im AWS X-Ray-Entwicklerhandbuch.

Themen

So funktioniert X-Ray mit Amazon S3

AWS X-Ray unterstützt die Trace-Kontextverbreitung für Amazon S3, sodass Sie End-to-End-Anforderungen ansehen können, während sie Ihre gesamte Anwendung durchlaufen. X-Ray aggregiert die Daten, die von einzelnen Services wie Amazon S3, AWS Lambda und Amazon EC2 generiert werden, sowie die vielen Ressourcen, aus denen sich Ihre Anwendung zusammensetzt. Es bietet Ihnen einen Überblick über die Leistung Ihrer Anwendung.

Amazon S3 lässt sich in X-Ray integrieren, um den Trace-Kontext zu verbreiten und Ihnen eine Anforderungskette mit Upstream- und Downstream-Knoten bereitzustellen. Wenn ein Upstream-Service einen korrekt formatierten Trace-Header in seine S3-Anforderung einfügt, wird der Trace-Header bei der Übermittlung von Ereignisbenachrichtigungen an Downstream-Services wie Lambda, Amazon SQS und Amazon SNS von Amazon S3 übergeben. Wenn diese Services aktiv in X-Ray integriert sind, sind sie in einer Anforderungskette verknüpft, um Ihnen die vollständigen Details Ihrer Amazon S3-Anforderungen bereitzustellen.

Um X-Ray-Trace-Header über Amazon S3 zu senden, müssen Sie eine formatierte X-Amzn-Trace-ID in Ihre Anforderungen aufnehmen. Sie können den Amazon-S3-Client auch mithilfe der AWS X-Ray-SDKs instrumentieren. Eine Liste der unterstützten SDKs finden Sie in der AWS-X-Ray-Dokumentation.

Service-Übersichten

In X-Ray-Service-Übersichten können Sie die Beziehungen zwischen Amazon S3 und anderen AWS-Services und -Ressourcen in Ihrer Anwendung nahezu in Echtzeit nachverfolgen. Um die End-to-End-Anforderungen in den X-Ray-Service-Übersichten einzusehen, können Sie in der die X-Ray-Konsole eine Übersicht der Verbindungen zwischen Amazon S3 und anderen Services anzeigen, die Ihre Anwendung verwendet. Sie können leicht erkennen, wo eine hohe Latenz auftritt, die Verteilung der Knoten für diese Services anzeigen und dann Details zu den verschiedenen Services und Pfaden aufrufen, die die Performance der Anwendung beeinträchtigen.

X-Ray-Analysen

Ebenso können Sie die X-Ray-Analysen-Konsole nutzen, um Traces zu analysieren, Metriken wie Latenz und Fehlerraten anzusehen und Erkenntnisse zu generieren, die Ihnen die Suche und Behebung von Fehlern erleichtern. In dieser Konsole werden auch Metriken wie durchschnittliche Latenz und Fehlerraten angezeigt. Weitere Informationen finden Sie unter AWS-X-Ray-Konsole im AWS-X-Ray-Entwicklerhandbuch.

Verfügbare Regionen

AWS X-Ray-Support für Amazon S3 ist in allen AWS X-Ray-Regionen verfügbar. Weitere Informationen finden Sie unter Amazon S3 und AWS-X-Ray im AWS-X-Ray-Entwicklerhandbuch.

Hosten einer statischen Website mit Amazon S3

Sie können Amazon S3 verwenden, um eine statische Website zu hosten. Auf einer statischen Website enthalten einzelne Webseiten statischen Inhalt. Sie könnten auch clientseitige Skripts enthalten.

Im Gegensatz dazu basiert eine dynamische Website auf einer serverseitigen Verarbeitung, einschließlich serverseitiger Skripts wie PHP, JSP oder ASP.NET. Amazon S3 unterstützt kein serverseitiges Scripting, AWS verfügt jedoch über andere Ressourcen zum Hosten dynamischer Websites. Um mehr über das Website-Hosting auf AWS zu erfahren, lesen Sie Web-Hosting.

> **Note**
>
> Sie können die AWS Amplify-Konsole verwenden, um eine Single-Page-Webanwendung zu hosten. Die AWS Amplify-Konsole unterstützt Single-Page-Anwendungen, die mit Single-Page-Anwendungs-Frameworks (z. B. React JS, Vue JS, Angular JS und Nuxt) und statischen Websitegeneratoren (z. B. Gatsby JS, React-Static, Jekyll und Hugo) entwickelt wurden. Weitere Informationen finden Sie unter Erste Schritte im Benutzerhandbuch für die AWS Amplify-Konsole.
>
> Die Amazon-S3-Website-Endpunkte unterstützen kein HTTPS. Wenn Sie HTTPS verwenden möchten, können Sie Amazon verwenden, CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird. Weitere Informationen finden Sie unter Wie verwende CloudFront ich , um HTTPS-Anfragen für meinen Amazon S3-Bucket zu bedienen? Um HTTPS mit einer benutzerdefinierten Domäne zu verwenden, siehe Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne.

Weitere Informationen zum Hosten einer statischen Website auf Amazon S3, einschließlich Anweisungen und step-by-step Anleitungen, finden Sie in den folgenden Themen.

Themen

- Website-Endpunkte
- Aktivieren des Website-Hostings
- Konfigurieren eines Indexdokuments
- Konfigurieren eines benutzerdefinierten Fehlerdokuments
- Festlegen von Berechtigungen für den Website-Zugriff

- (Optional) Protokollieren des Webdatenverkehrs
- (Optional) Konfigurieren einer Webseitenumleitung

Website-Endpunkte

Wenn Sie Ihren Bucket als statische Website konfigurieren, steht die Website an dem für die AWS-Region-spezifischen Website-Endpunkt des Buckets zur Verfügung. Website-Endpunkte unterscheiden sich von den Endpunkten, von denen aus Sie REST-API-Anfragen senden. Weitere Informationen zu den Unterschieden zwischen den Endpunkten finden Sie unter Wichtige Unterschiede zwischen einem Website-Endpunkt und einem REST-API-Endpunkt.

Je nach Region weisen Ihre Amazon-S3-Website-Endpunkte eines der beiden folgenden Formate auf.

- s3-website dash (-) Region - `http://bucket-name.s3-website-Region.amazonaws.com`
- s3-website dot (.) Region - `http://bucket-name.s3-website.Region.amazonaws.com`

Diese URLs geben ein Standard-Indexdokument zurück, das Sie für die Website konfiguriert haben: Eine vollständige Liste der Amazon-S3-Website-Endpunkte finden Sie unter Amazon-S3-Website-Endpunkte.

> **Note**
>
> Um die Sicherheit Ihrer statischen Amazon S3-Websites zu erhöhen, werden die Amazon S3-Website-Endpunktdomänen (z. B. s3website-us-east--1.amazonaws.com oder s3-website.ap-south-1.amazonaws.com) in der Public Suffix List (PSL) registriert. Aus Sicherheitsgründen empfehlen wir Ihnen, Cookies mit einem `__Host--`Präfix zu verwenden, falls Sie jemals sensible Cookies im Domain-Namen für statische Amazon-S3-Webistes einrichten müssen. Diese Vorgehensweise hilft Ihnen dabei, Ihre Domain vor CSRF-Versuchen (Cross-Site Request Forgery Attempts, Anforderungsfälschung zwischen Websites) zu schützen. Weitere Informationen finden Sie auf der Set-Cookie-Seite im Mozilla Developer Network.

Wenn Ihre Website öffentlich sein soll, müssen Sie Ihren gesamten Inhalt öffentlich lesbar machen, damit Ihre Kunden am Website-Endpunkt darauf zugreifen können. Weitere Informationen finden Sie unter Festlegen von Berechtigungen für den Website-Zugriff.

> **Important**
>
> Amazon S3 Website-Endpunkte unterstützen nicht HTTPS oder Zugriffspunkte. Wenn Sie HTTPS verwenden möchten, können Sie Amazon verwenden, CloudFront um eine statische Website bereitzustellen, die auf Amazon S3 gehostet wird. Weitere Informationen finden Sie unter Wie verwende ich , CloudFront um HTTPS-Anfragen für meinen Amazon S3-Bucket zu bedienen? Um HTTPS mit einer benutzerdefinierten Domäne zu verwenden, siehe Configuring a static website using a custom domain registered with Route 53 (Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne).
>
> Buckets mit Zahlung durch den Auftraggeber erlauben keinen Zugriff über den Website-Endpunkt. Jede Anforderung an einen solchen Bucket erhält die Antwort 403 Access Denied (403 Zugriff verweigert). Weitere Informationen finden Sie unter Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung.

Themen

Website-Endpunkt-Beispiele

Die folgenden Beispiele zeigen, wie Sie auf einen Amazon-S3-Bucket zugreifen können, der als statische Website konfiguriert ist.

Example – Anfordern eines Objekts auf Stammebene

Um ein bestimmtes Objekt anzufordern, das auf der Stammebene im Bucket gespeichert ist, verwenden Sie die folgende URL-Struktur.

```
http://bucket-name.s3-website.Region.amazonaws.com/object-name
```

Die folgende URL fordert beispielsweise das Objekt `photo.jpg` an, das auf der Stammebene im Bucket gespeichert ist.

```
http://example-bucket.s3-website.us-west-2.amazonaws.com/photo.jpg
```

Example – Anfordern eines Objekts in einem Präfix

Um ein Objekt anzufordern, das in einem Ordner in Ihrem Bucket gespeichert ist, verwenden Sie diese URL-Struktur.

```
http://bucket-name.s3-website.Region.amazonaws.com/folder-name/object-name
```

Die folgende URL fordert das `docs/doc1.html`-Objekt in Ihrem Bucket an.

```
http://example-bucket.s3-website.us-west-2.amazonaws.com/docs/doc1.html
```

Hinzufügen eines DNS-CNAME

Wenn Sie eine registrierte Domäne haben, können Sie einen DNS CNAME-Eintrag hinzufügen, der auf den Amazon-S3-Website-Endpunkt verweist. Wenn Sie beispielsweise die Domäne `www.example-bucket.com` registriert haben, könnten Sie den Bucket `www.example-bucket.com` erstellen und einen DNS CNAME-Datensatz hinzufügen, der auf `www.example-bucket.com.s3-website.Region.amazonaws.com` verweist. Alle Anfragen an `http://www.example-bucket.com` werden an `www.example-bucket.com.s3-website.Region.amazonaws.com` weitergeleitet.

Weitere Informationen finden Sie unter Anpassen von Amazon-S3-URLs mit CNAME-Einträgen.

Verwenden einer benutzerdefinierten Domäne mit Route 53

Anstatt über einen Amazon-S3-Website-Endpunkt auf die Website zuzugreifen, können Sie beispielsweise Ihre eigene Domäne verwenden, die bei Amazon Route 53 registriert ist, um Ihre Inhalte bereitzustellen, z. B, `example.com`. Sie können Amazon S3 mit Route 53 verwenden, um eine Website in der Root-Domäne zu hosten. Wenn Sie beispielsweise die Root-Domäne `example.com` haben und Ihre Website auf Amazon S3 hosten, können Ihre Website-Besucher von ihrem Browser aus auf die Seite zugreifen, indem sie `http://www.example.com` oder `http://example.com` eingeben.

Ein Beispiel-Walkthrough finden Sie unte Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne.

Wichtige Unterschiede zwischen einem Website-Endpunkt und einem REST-API-Endpunkt

Der Amazon-S3-Website-Endpunkt ist auf den Zugriff über einen Webbrowser optimiert. In der folgenden Tabelle werden die wichtigsten Unterschiede zwischen einem REST-API-Endpunkt und einem Website-Endpunkt zusammengefasst.

Wichtiger Unterschied	REST-API-Endpunkt	Website-Endpunkt
Zugriffskontrolle	Unterstützt öffentlichen und privaten Inhalt	Unterstützt nur öffentlich lesbaren Inhalt
Verarbeiten von Fehlermeldungen	Gibt eine XML-formatierte Fehlermeldung zurück	Gibt ein HTML-Dokument zurück
Unterstützung einer Umleitung	Nicht zutreffend	Unterstützt Umleitungen auf Objekt- und auf Bucket-Ebene
Unterstützte Anfragen	Unterstützt alle Bucket- und Objekt-Vorgänge	Unterstützt nur GET- und HEAD-Anforderungen für Objekte
Reagiert auf GET- und HEAD-Anfragen an der Root eines Buckets	Gibt eine Liste der Objektschlüssel im Bucket zurück.	Gibt das in der Website-Konfiguration angegebene Indexdokument zurück.
Support von Secure Sockets Layer (SSL)	Unterstützt SSL-Verbindungen	Unterstützt keine SSL-Verbindungen.

Eine vollständige Liste der Amazon-S3-Endpunkte finden Sie unter Amazon-S3-Endpunkte und -Kontingente in der Allgemeine AWS-Referenz.

Aktivieren des Website-Hostings

Wenn Sie einen Bucket als statische Website konfigurieren, müssen Sie das statische Website-Hosting aktivieren, ein Indexdokument konfigurieren und Berechtigungen festlegen.

Sie können statisches Website-Hosting mit der Amazon-S3-Konsole, der REST-API, den AWS-SDKs, der AWS CLI oder AWS CloudFormation aktivieren.

Informationen zum Konfigurieren Ihrer Website mit einer benutzerdefinierten Domäne finden Sie unter Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne.

Verwenden der S3-Konsole

So aktivieren Sie das statische Website-Hosting

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie das Hosting statischer Websites aktivieren wollen.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Static website hosting (Hosting statischer Websites) Edit (Bearbeiten) aus.
5. Wählen Sie Use this bucket to host a website (Diesen Bucket zum Hosten einer Website verwenden).
6. Wählen Sie unter Static website hosting (Hosting statischer Websites) die Option Enable (Aktivieren) aus
7. Geben Sie unter Index document (Index-Dokument) den Dateinamen des Index-Dokuments ein, der typischerweise `index.html` ist.

   Der Name des Indexdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Indexdokuments übereinstimmen, das Sie in den S3-Bucket hochladen möchten. Wenn Sie Ihren Bucket für das Hosting von Websites konfigurieren, müssen Sie ein Indexdokument angeben. Amazon S3 gibt dieses Indexdokument zurück, wenn Anfragen an die Root-Domäne oder einen der Unterordner gestellt werden. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments.
8. Um ein eigenes benutzerdefiniertes Fehlerdokument für Fehler der Klasse 4XX bereitzustellen, geben Sie unter Fehlerdokument den Dateinamen des benutzerdefinierten Fehlerdokuments ein.

Der Name des Fehlerdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Fehlerdokuments übereinstimmen, das Sie in Ihren S3-Bucket hochladen möchten. Wenn Sie kein benutzerdefiniertes Fehlerdokument angeben und ein Fehler auftritt, wird von Amazon S3 ein Standard-HTML-Fehlerdokument zurückgegeben. Weitere Informationen finden Sie unter Konfigurieren eines benutzerdefinierten Fehlerdokuments.

9. (Optional) Wenn Sie fortschrittliche Umleitungsregeln angeben möchten, geben Sie unter Redirection rules (Umleitungsregeln) JSON zur Beschreibung der Regeln ein.

   Beispielsweise können Sie bedingt Anfragen abhängig von bestimmten Objektschlüsselnamen oder Präfixen in der Anfrage weiterleiten. Weitere Informationen finden Sie unter Konfigurieren von Umleitungsregeln für die Verwendung von fortschrittliche bedingten Umleitungen.

10. Wählen Sie Save Changes (Änderungen speichern).

    Amazon S3 ermöglicht statisches Website-Hosting für Ihren Bucket. Unten auf der Seite sehen Sie unter Static website hosting (Hosting statischer Websites) den Website-Endpunkt für Ihren Bucket.

11. Notieren Sie unter Static website hosting (Statisches Website-Hosting) den Wert für Endpoint (Endpunkt).

    Der Endpoint (Endpunkt) ist der Amazon-S3-Website-Endpunkt für Ihren Bucket. Nachdem Sie den Bucket als statische Website konfiguriert haben, können Sie diesen Endpunkt verwenden, um Ihre Website zu testen.

Verwenden der REST-API

Weitere Informationen zum Senden von direkten REST-Anfragen zur Aktivierung des statischen Website-Hostings finden Sie in den folgenden Abschnitten der Amazon Simple Storage Service API-Referenz:

- PUT Bucket-Website
- GET Bucket-Website
- DELETE Bucket-Website

Verwenden der AWS-SDKs

Um eine statische Website auf Amazon S3 zu hosten, konfigurieren Sie einen Amazon-S3-Bucket für ein Website-Hosting und laden dann Ihren Website-Inhalt in den Bucket hoch. Sie können die AWS SDKs auch verwenden, um die Websitekonfiguration programmgesteuert zu erstellen, zu aktualisieren und zu löschen. Die SDKs stellen Wrapper-Klassen für die Amazon-S3-REST-APIs bereit. Falls in Ihrer Anwendung erforderlich, können Sie auch direkt von Ihrer Anwendung aus REST-API-Anfragen senden.

.NET

Das folgende Beispiel veranschaulicht, wie Sie mit AWS SDK for .NET die Website-Konfiguration für einen Bucket verwalten. Um einem Bucket eine Website-Konfiguration hinzuzufügen, geben Sie einen Bucket-Namen und eine Website-Konfiguration an. Die Website-Konfiguration muss ein Indexdokument enthalten und kann optional ein Fehlerdokument enthalten. Diese Dokumente müssen im Bucket gespeichert sein. Weitere Informationen finden Sie unter PUT Bucket-Website. Weitere Informationen zur Amazon-S3-Website-Funktion finden Sie unter Hosten einer statischen Website mit Amazon S3.

Das folgende C#-Code-Beispiel fügt dem angegebenen Bucket eine Website-Konfiguration hinzu. Die Konfiguration gibt die Namen für das Indexdokument und das Fehlerdokument an. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class WebsiteConfigTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string indexDocumentSuffix = "*** index object key ***"; //
 For example, index.html.
        private const string errorDocument = "*** error object key ***"; // For
 example, error.html.
        // Specify your bucket region (an example region is shown).
```

```
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            AddWebsiteConfigurationAsync(bucketName, indexDocumentSuffix,
errorDocument).Wait();
        }

        static async Task AddWebsiteConfigurationAsync(string bucketName,
                                            string indexDocumentSuffix,
                                            string errorDocument)
        {
            try
            {
                // 1. Put the website configuration.
                PutBucketWebsiteRequest putRequest = new PutBucketWebsiteRequest()
                {
                    BucketName = bucketName,
                    WebsiteConfiguration = new WebsiteConfiguration()
                    {
                        IndexDocumentSuffix = indexDocumentSuffix,
                        ErrorDocument = errorDocument
                    }
                };
                PutBucketWebsiteResponse response = await
client.PutBucketWebsiteAsync(putRequest);

                // 2. Get the website configuration.
                GetBucketWebsiteRequest getRequest = new GetBucketWebsiteRequest()
                {
                    BucketName = bucketName
                };
                GetBucketWebsiteResponse getResponse = await
client.GetBucketWebsiteAsync(getRequest);
                Console.WriteLine("Index document: {0}",
getResponse.WebsiteConfiguration.IndexDocumentSuffix);
                Console.WriteLine("Error document: {0}",
getResponse.WebsiteConfiguration.ErrorDocument);
            }
            catch (AmazonS3Exception e)
            {
```

```
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

PHP

Das folgende PHP-Beispiel fügt dem angegebenen Bucket eine Website-Konfiguration hinzu. Die Methode `create_website_config` stellt explizit die Namen des Indexdokuments und des Fehlerdokuments bereit. Das Beispiel ruft auch die Website-Konfiguration ab und gibt die Antwort aus. Weitere Informationen zur Amazon-S3-Website-Funktion finden Sie unter Hosten einer statischen Website mit Amazon S3.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);


// Add the website configuration.
$s3->putBucketWebsite([
    'Bucket'                => $bucket,
    'WebsiteConfiguration'  => [
        'IndexDocument' => ['Suffix' => 'index.html'],
        'ErrorDocument' => ['Key' => 'error.html']
    ]
]);
```

```
// Retrieve the website configuration.
$result = $s3->getBucketWebsite([
    'Bucket' => $bucket
]);
echo $result->getPath('IndexDocument/Suffix');

// Delete the website configuration.
$s3->deleteBucketWebsite([
    'Bucket' => $bucket
]);
```

Verwendung der AWS CLI

Weitere Informationen zur Verwendung der AWS CLI zur Konfiguration eines S3-Buckets als statische Website finden Sie unter Website in der AWS CLI-Befehlsreferenz.

Als Nächstes müssen Sie das Indexdokument konfigurieren und Berechtigungen festlegen. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments und Festlegen von Berechtigungen für den Website-Zugriff.

Optional können Sie auch ein Fehlerdokument, eineWebdatenverkehrs-Protokollierung oder eine Weiterleitung konfigurieren.

Konfigurieren eines Indexdokuments

Wenn Sie Website-Hosting aktivieren, müssen Sie auch ein Indexdokument konfigurieren und hochladen. Ein Indexdokument ist eine Webseite, die Amazon S3 zurückgibt, wenn eine Anforderung für die Root einer Website oder einen Unterordner erfolgt. Wenn ein Benutzer beispielsweise `http://www.example.com` in den Browser eingibt, fordert er keine spezifische Seite an. In diesem Fall stellt Amazon S3 das Indexdokument bereit, das manchmal auch als Standardseite bezeichnet wird.

Wenn Sie das statische Website-Hosting für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B, `index.html`). Nachdem Sie das statische Website-Hosting für Ihren Bucket aktiviert haben, laden Sie eine HTML-Datei mit dem Namen des Indexdokuments in Ihren Bucket hoch.

Der abschließend Schrägstrich der Root-Level-URL ist optional. Wenn Sie beispielsweise Ihre Website mit `index.html` als Indexdokument konfigurieren, gibt jede der folgenden URLs `index.html` zurück.

```
http://example-bucket.s3-website.Region.amazonaws.com/
http://example-bucket.s3-website.Region.amazonaws.com
```

Weitere Informationen zu Amazon-S3-Website-Endpunkten finden Sie unter Website-Endpunkte.

Indexdokument und Ordner

In Amazon S3 ist ein Bucket ein eindimensionaler Container mit Objekten. Er bietet keine hierarchische Organisation wie das Dateisystem auf Ihrem Computer. Sie können jedoch mit Objektschlüsselnamen, die eine Ordnerstruktur implizieren, eine logische Hierarchie erstellen.

Betrachten wir beispielsweise einen Bucket mit drei Objekten, die die folgenden Schlüsselnamen besitzen. Obwohl diese nicht in einer physischen Hierarchie gespeichert sind, können Sie aus den Schlüsselnamen die folgende logische Ordnerstruktur ableiten:

- `sample1.jpg` – Das Objekt befindet sich im Stammverzeichnis des Buckets.
- `photos/2006/Jan/sample2.jpg` – Das Objekt befindet sich im Unterordner `photos/2006/Jan`.
- `photos/2006/Feb/sample3.jpg` – Das Objekt befindet sich im Unterordner `photos/2006/Feb`.

In der Amazon-S3-Konsole können Sie auch einen Ordner in einem Bucket erstellen. Sie können beispielsweise einen Ordner mit dem Namen erstelle `photos`. Sie können Objekte in den Bucket hochladen, oder in den `photos`-Ordner im Bucket. Wenn Sie dem Bucket das Objekt `sample.jpg` hinzufügen, ist der Schlüsselname `sample.jpg`. Wenn Sie das Objekt in den Ordner `photos` hochladen, ist der Objektschlüsselname `photos/sample.jpg`.

Wenn Sie eine Ordnerstruktur in Ihrem Bucket erstellen, muss es auf jeder Ebene ein Indexdokument geben. In jedem Ordner muss das Indexdokument den gleichen Namen haben, `index.html`. Wenn ein Benutzer eine URL angibt, die an eine Ordnersuche erinnert, bestimmt das Vorhandensein oder Fehlen eines abschließenden Schrägstrichs das Verhalten der Website. Die folgende URL beispielsweise mit einem abschließenden Schrägstrich gibt das Indexdokument `photos/index.html` zurück.

```
http://bucket-name.s3-website.Region.amazonaws.com/photos/
```

Wenn Sie den abschließenden Schrägstrich aus der obigen URL weglassen, sucht Amazon S3 zuerst im Bucket nach dem Objekt `photos`. Wird das Objekt `photos` nicht gefunden, sucht es nach einem Indexdokument, `photos/index.html`. Wird dieses Dokument gefunden, gibt Amazon S3 eine `302 Found`-Meldung zurück und verweist auf den `photos/`-Schlüssel. Für nachfolgende Anforderungen für `photos/` gibt Amazon S3 `photos/index.html` zurück. Wird das Indexdokument nicht gefunden, gibt Amazon S3 einen Fehler zurück.

Konfigurieren eines Indexdokuments

Gehen Sie wie folgt vor, um ein Indexdokument mit der S3-Konsole zu konfigurieren. Sie können ein Indexdokument auch über die REST-API, die AWS-SDKs, die AWS CLI oder AWS CloudFormation konfigurieren.

> **Note**
>
> In einem Bucket mit aktivierter Versionsverwaltung können Sie mehrere Kopien der Datei „`index.html`“ hochladen, es wird jedoch nur die neueste Version verwendet. Weitere Informationen zur Verwendung der S3-Versionsverwaltung finden Sie unter Verwenden der Versioning in S3-Buckets.

Wenn Sie das statische Website-Hosting für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B, **`index.html`**). Nachdem Sie das Hosting statischer Websites für den Bucket aktiviert haben, laden Sie eine HTML-Datei mit diesem Indexdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie das Indexdokument

1. Erstellen Sie eine Datei `index.html`.

   Wenn Sie nicht über eine Datei `index.html` verfügen, können Sie mit dem folgenden HTML-Code eine Datei erstellen:

   ```
   <html xmlns="http://www.w3.org/1999/xhtml" >
   <head>
       <title>My Website Home Page</title>
   ```

```
</head>
<body>
  <h1>Welcome to my website</h1>
  <p>Now hosted on Amazon S3!</p>
</body>
</html>
```

2. Speichern Sie die Indexdatei lokal.

   Der Dateiname des Indexdokuments muss genau mit dem Namen des Indexdokuments übereinstimmen, den Sie im Dialogfeld Static website hosting (Statisches Website-Hosting) eingeben. Beim Namen des Indexdokuments wird die Groß- und Kleinschreibung berücksichtigt. Wenn Sie beispielsweise im Dialogfeld Static website hosting (Statisches Website-Hosting) `index.html` als den Namen des Index document (Indexdokuments) eingeben, muss der Dateiname des Indexdokuments ebenfalls `index.html` und nicht `Index.html` lauten.

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.

5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Indexdokuments ein (z. B. `index.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings.

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.

6. Führen Sie einen der folgenden Schritte aus, um das Indexdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Indexdatei per Drag & Drop in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   step-by-step Anweisungen finden Sie unter Objekte hochladen.

7. (Optional) Laden Sie andere Website-Inhalte in Ihren Bucket hoch.

Als Nächstes müssen Sie Berechtigungen für den Websitezugriff festlegen. Weitere Informationen finden Sie unter Festlegen von Berechtigungen für den Website-Zugriff.

Optional können Sie auch ein Fehlerdokument, eineWebdatenverkehrs-Protokollierung oder eine Weiterleitung konfigurieren.

Konfigurieren eines benutzerdefinierten Fehlerdokuments

Nachdem Sie Ihren Bucket als statische Website konfiguriert haben, gibt Amazon S3 ein HTML-Fehlerdokument zurück, wenn ein Fehler auftritt. Sie können Ihren Bucket optional mit einem benutzerdefinierten Fehlerdokument konfigurieren, sodass Amazon S3 bei Auftreten eines Fehlers dieses Dokument zurückgibt.

> **Note**
>
> Tritt ein Fehler auf, zeigen einige Browser beim Auftreten eines Fehlers ihre eigene Fehlermeldung an und ignorieren das Fehlerdokument, das Amazon S3 zurückgibt. Tritt beispielsweise der Fehler HTTP 404 Not Found auf, könnte Google Chrome seine eigene Fehlermeldung anzeigen und das von Amazon S3 zurückgegebene Fehlerdokument ignorieren.

Themen

Amazon S3 HTTP-Antwortcodes

Die folgende Tabelle listet die Teilmenge der HTTP-Antwortcodes auf, die Amazon S3 zurückgibt, wenn ein Fehler auftritt.

HTTP-Fehlercode	Beschreibung
301 Moved Permanently (301 Dauerhaft verschoben)	Wenn ein Benutzer eine Anforderung direkt an den Amazon-S3-Website-Endpunkt sendet (`http://s3-website.` *`Region`*`.amazonaws.com/`), gibt Amazon S3 die Antwort 301 Moved Permanently zurück und leitet diese Anforderungen auf `https://aws.amazon.com/s3/` um.
302 Found (302 Gefunden)	

HTTP-Fehlercode	Beschreibung
	Wenn Amazon S3 eine Anforderung für einen Schlüssel `x`, `http://bucket-name .s3-website. Region.amazonaws.com/x` ohne nachfolgenden Schrägstrich erhält, sucht es zuerst nach dem Objekt mit dem Schlüsselnamen `x`. Wenn das Objekt nicht gefunden wird, stellt Amazon S3 fest, dass die Anfrage für den Unterordner `x` vorgesehen ist, und leitet die Anfrage um, indem es einen Schrägstrich am Ende einfügt. Es gibt 302 Found zurück.
304 Not Modified (304 Nicht verändert)	Amazon S3 verwendet die Anforderungs-Header `If-Modified-Since` , `If-Unmodified-Since` , `If-Match` und/oder `If-None-Match` , um festzustellen, ob das angeforderte Objekt dasselbe ist, wie das, das der Client im Cache vorhält. Ist das Objekt dasselbe, gibt der Website-Endpunkt die Antwort 304 Not Modified (304 Nicht verändert) zurück.
400 Malformed Request (400 Falsch formatierte Anfrage)	Der Website-Endpunkt antwortet mit 400 Malformed Request (400 Falsch formatierte Anfrage), wenn ein Benutzer versucht, über einen inkorrekten regionalen Endpunkt auf einen Bucket zuzugreifen.
403 Forbidden	Der Website-Endpunkt antwortet mit 403 Forbidden (403 Verboten), wenn ein Benutzer auf ein Objekt zugreifen will, das nicht öffentlich lesbar ist. Der Objekteigentümer muss zuerst das Objekt über eine Bucket-Richtlinie oder eine ACL öffentlich lesbar machen.

HTTP-Fehlercode	Beschreibung
404 Not Found (404 Nicht gefunden)	Der Website-Endpunkt gibt aus den folgenden Gründen 404 Not Found (404 Nicht gefunden) zurück: • Amazon S3 stellt fest, dass die URL der Website auf einen nicht existiere nden Objektschlüssel verweist. • Amazon S3 geht davon aus, dass die Anforderung für ein Indexdokument gilt, das nicht existiert. • Ein in der URL angegebener Bucket ist nicht vorhanden. • Ein in der URL angegebener Bucket ist vorhanden, aber nicht als Website konfiguriert. Sie können ein benutzerdefiniertes Objekt erstellen, das für 404 Not Found (404 Nicht gefunden) zurückgegeben wird. Stellen Sie sicher, dass das Dokument in den als Website konfigurierten Bucket hochgeladen ist, und dass die Konfiguration für das Website-Hosting darauf ausgelegt ist, das Dokument zu verwenden. Weitere Informationen darüber, wie Amazon S3 die URL als Anforderung für ein Objekt oder ein Indexdokument interpretiert, finden Sie unter Konfiguri eren eines Indexdokuments.
500 Service Error (500 Servicefe hler)	Der Website-Endpunkt reagiert mit 500 Service Error (500 Servicefehler), wenn ein interner Serverfehler auftritt.
503 Service Unavailable (503 Service nicht verfügbar)	Der Website-Endpunkt antwortet mit 503 Service Unavailable, wenn Amazon S3 feststellt, dass Sie Ihre Anforderungsrate reduzieren müssen.

Für jeden dieser Fehler gibt Amazon S3 eine vordefinierte HTML-Meldung zurück. Nachfolgend sehen Sie ein Beispiel für eine HTML-Meldung, die für eine 403 Forbidden (403 Verboten)-Nachricht zurückgegeben wird.

Konfigurieren eines benutzerdefinierten Fehlerdokuments

Wenn Sie Ihren Bucket als statische Website konfigurieren, können Sie ein benutzerdefiniertes Fehlerdokument bereitstellen, das eine benutzerfreundliche Fehlermeldung und zusätzliche Hilfe enthält. Amazon S3 gibt Ihr benutzerdefiniertes Fehlerdokument nur für Fehlercodes der Klasse HTTP 4XX zurück.

Um ein benutzerdefiniertes Fehlerdokument über die S3-Konsole zu konfigurieren, führen Sie die folgenden Schritte aus. Sie können ein Fehlerdokument auch über die REST-API, die AWS-SDKs, die AWS CLI oder AWS CloudFormation konfigurieren. Weitere Informationen finden Sie hier:

- PutBucketWebsite in der API-Referenz zu Amazon Simple Storage Service
- AWS::S3::Bucket WebsiteConfiguration im AWS CloudFormation-Benutzerhandbuch
- put-bucket-website in der AWS CLI Befehlsreferenz

Wenn Sie das Hosting statischer Websites für Ihren Bucket aktivieren, geben Sie den Namen des Fehlerdokuments ein (z. B, **`404.html`**). Nachdem das Hosting statischer Websites für den Bucket aktiviert wurde, laden Sie eine HTML-Datei mit diesem Fehlerdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie ein Fehlerdokument

1. Erstellen Sie ein Fehlerdokument, z. B. `404.html`.

2. Speichern Sie die Fehlerdokumentdatei lokal.

   Der Name des Fehlerdokuments unterscheidet zwischen Groß- und Kleinschreibung und muss genau mit dem Namen übereinstimmen, den Sie beim Aktivieren des statischen Website-Hostings eingeben. Wenn Sie beispielsweise `404.html` im Dialogfeld Hosten einer statischen Website als Namen des Fehlerdokuments eingeben, muss der Dateiname des Fehlerdokuments ebenfalls `404.html` lauten.
3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.
5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Fehlerdokuments ein (z. B, `404.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings und Konfigurieren eines benutzerdefinierten Fehlerdokuments.

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.
6. Führen Sie einen der folgenden Schritte aus, um das Fehlerdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Fehlerdokumentdatei in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   step-by-step Anweisungen finden Sie unter Objekte hochladen.

Festlegen von Berechtigungen für den Website-Zugriff

Wenn Sie einen Bucket als statische Website konfigurieren und Ihre Website öffentlich sein soll, können Sie öffentlichen Lesezugriff gewähren. Um Ihren Bucket öffentlich lesbar zu machen, müssen Sie die Block-Einstellungen für den öffentlichen Zugriff für den Bucket deaktivieren und eine Bucket-Richtlinie schreiben, die öffentlichen Lesezugriff gewährt. Wenn Ihr Bucket Objekte enthält, die nicht im Besitz des Bucket-Eigentümers sind, müssen Sie möglicherweise auch eine Objekt-Zugriffskontrollliste (Access Control List, ACL) hinzufügen, die jedem Benutzer Lesezugriff erteilt.

Wenn Sie die Block Public Access-Einstellungen für Ihren Bucket nicht deaktivieren möchten, Ihre Website aber trotzdem öffentlich sein soll, können Sie eine Amazon- CloudFront Verteilung erstellen,

um Ihre statische Website bereitzustellen. Weitere Informationen finden Sie unter Beschleunigen Ihrer Website mit Amazon CloudFront oder Verwenden einer Amazon- CloudFront Verteilung zur Bereitstellung einer statischen Website im Amazon Route 53-Entwicklerhandbuch.

> **Note**
>
> Am Website-Endpunkt gibt Amazon S3 den HTTP-Antwortcode `404 (Not Found)` zurück, wenn ein Benutzer ein nicht existierendes Objekt anfordert. Wenn das Objekt existiert, aber Sie keine Leseberechtigungen dafür erteilt haben, gibt der Website-Endpunkt den HTTP-Antwortcode `403 (Access Denied)` zurück. Der Benutzer kann aus dem Antwortcode ableiten, ob ein bestimmtes Objekt existiert. Wenn Sie dieses Verhalten nicht wünschen, sollten Sie den Website-Support für Ihren Bucket nicht aktivieren.

Themen

Schritt 1: Bearbeiten der S3 Block Public Access-Einstellungen

Wenn Sie einen bestehenden Bucket als statische Website konfigurieren möchten, die einen öffentlichen Zugriff bietet, müssen Sie die Block Public Access-Einstellungen für diesen Bucket bearbeiten. Möglicherweise müssen Sie auch Ihre Einstellungen für Block Public Access auf Kontoebene bearbeiten. Amazon S3 wendet die restriktivste Kombination der Block Public Access-Einstellungen auf Bucket- und Kontoebene an.

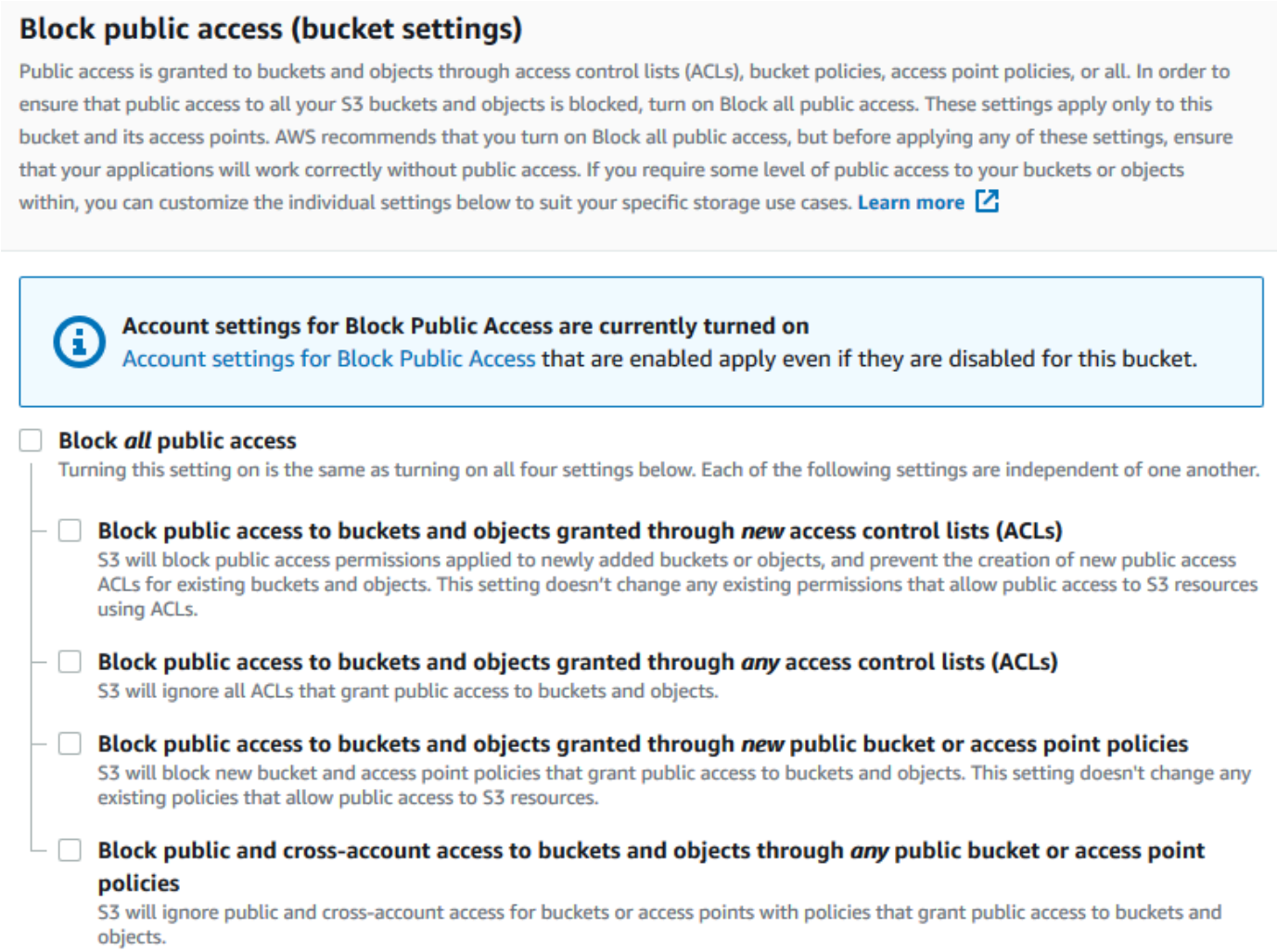
Wenn Sie beispielsweise den öffentlichen Zugriff für einen Bucket erlauben, aber den gesamten öffentlichen Zugriff auf Kontoebene blockieren, blockiert Amazon S3 den öffentlichen Zugriff auf den Bucket weiter. In diesem Szenario müssten Sie Ihre Block Public Access-Einstellungen auf Bucket- und Kontoebene bearbeiten. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher.

Standardmäßig blockiert Amazon S3 den öffentlichen Zugriff auf Ihr Konto und Ihre Buckets. Wenn Sie einen Bucket verwenden möchten, um eine statische Website zu hosten, können Sie diese Schritte verwenden, um Ihre Einstellungen für Block Public Access zu bearbeiten:

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Permissions (Berechtigungen).
4. Wählen Sie unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)), die Option Edit (Bearbeiten).
5. Löschen Sie Block all public access (Gesamten öffentlichen Zugriff blockieren) und wählen Sie Save (Speichern).

> **Warning**
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.



Amazon S3 deaktiviert die Block Public Access-Einstellungen für Ihren Bucket. Um eine öffentliche, statische Website zu erstellen, müssen Sie möglicherweise auch die Block Public Access-Einstellungen für Ihr Konto bearbeiten, bevor Sie eine Bucket-Richtlinie hinzufügen. Wenn Kontoeinstellungen für Block Public Access derzeit aktiviert sind, wird unter Block public access (bucket settings) (Öffentlichen Zugriff blockieren (Bucket-Einstellungen)) ein Hinweis angezeigt.

Schritt 2: Hinzufügen einer Bucket-Richtlinie

Um die Objekte in Ihrem Bucket öffentlich lesbar zu machen, müssen Sie eine Bucket-Richtlinie schreiben, die jedem die `s3:GetObject`-Berechtigung erteilt.

Nachdem Sie die Einstellungen für S3 Block Public Access bearbeitet haben, können Sie eine Bucket-Richtlinie hinzufügen, um öffentlichen Lesezugriff auf den Bucket zu gewähren. Wenn Sie öffentlichen Lesezugriff gewähren, kann jeder im Internet auf Ihren Bucket zugreifen.

> **Important**
>
> Die zuvor genannte Richtlinie ist nur ein Beispiel und erlaubt Vollzugriff auf die Inhalte Ihres Buckets. Bevor Sie mit diesem Schritt fortfahren, lesen Sie den Abschnitt Wie kann ich die Dateien in meinem Amazon-S3-Bucket sichern?, um sicherzustellen, dass Sie die bewährten Methoden zum Sichern der Dateien in Ihrem S3-Bucket und die Risiken in Zusammenhang mit der Gewährung von öffentlichem Zugriff kennen.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Permissions (Berechtigungen).
3. Wählen Sie unter Bucket Policy (Bucket-Richtlinie) Edit (Bearbeiten).
4. Um öffentlichen Lesezugriff auf Ihre Website zu gewähren, kopieren Sie die folgende Bucket-Richtlinie und fügen Sie sie in den Bucket policy editor (Bucket-Richtlinieneditor) ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::Bucket-Name/*"
            ]
        }
    ]
}
```

5. Aktualisieren Sie den `Resource` zu Ihrem Bucket-Namen.

In der obigen Beispiel-Bucket-Richtlinie ist *Bucket-Name* ein Platzhalter für den Bucket-Namen. Um diese Bucket-Richtlinie mit Ihrem eigenen Bucket zu verwenden, müssen Sie diesen Namen so aktualisieren, dass er mit Ihrem Bucket übereinstimmt.

6. Wählen Sie Save Changes (Änderungen speichern).

   Es wird eine Meldung angezeigt, die darauf hinweist, dass die Bucket-Richtlinie erfolgreich hinzugefügt wurde.

   Wenn die Fehlermeldung `Policy has invalid resource` angezeigt wird, bestätigen Sie, dass der Bucket-Name in der Bucket-Richtlinie mit Ihrem Bucket-Namen übereinstimmt. Informationen zum Hinzufügen einer Bucket-Richtlinie finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu?

   Wenn Sie eine Fehlermeldung erhalten und die Bucket-Richtlinie nicht speichern können, überprüfen Sie Ihr Konto und die Bucket-Einstellungen für Block Public Access, um zu bestätigen, dass Sie den öffentlichen Zugriff auf den Bucket zulassen.

Objektzugriffskontrolllisten

Sie können eine Bucket-Richtlinie verwenden, um Ihren Objekten öffentliche Leserechte zu erteilen. Die Bucket-Richtlinie gilt jedoch nur für Objekte, die sich im Besitz des Bucket-Eigentümers befinden. Wenn Ihr Bucket Objekte enthält, die nicht dem Bucket-Eigentümer gehören, sollte der Bucket-Eigentümer die Objekt-Access-Control-List (ACL) verwenden, um öffentlichen Lesezugriff auf diese Objekte zu gewähren.

S3 Object Ownership ist eine Amazon-S3-Einstellung auf Bucket-Ebene, mit der Sie sowohl die Eigentümerschaft von den Objekten steuern können, die in Ihre Buckets hochgeladen werden, als auch ACLs deaktivieren oder aktivieren können. Standardmäßig ist die Objekteigentümerschaft auf die Einstellung „Vom Bucket-Eigentümer erzwungen“ festgelegt und alle ACLs sind deaktiviert. Wenn ACLs deaktiviert sind, besitzt der Bucket-Eigentümer alle Objekte im Bucket und verwaltet den Zugriff darauf ausschließlich mithilfe von Zugriffsverwaltungsrichtlinien.

Die meisten modernen Anwendungsfälle in Amazon S3 erfordern keine ACLs mehr. Wir empfehlen Ihnen, ACLs deaktiviert zu lassen, außer unter ungewöhnlichen Umständen, in denen Sie den Zugriff für jedes Objekt einzeln steuern müssen. Wenn ACLs deaktiviert sind, können Sie mithilfe von Richtlinien den Zugriff auf alle Objekte in Ihrem Bucket steuern, unabhängig davon, wer die Objekte

in Ihren Bucket hochgeladen hat. Weitere Informationen finden Sie unter Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket..

> **Important**
>
> Wenn Ihr Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet, müssen Sie Richtlinien verwenden, um Zugriff auf Ihren Bucket und die darin enthaltenen Objekte zu gewähren. Wenn die Einstellung „Vom Bucket-Eigentümer erzwungen“ aktiviert ist, schlagen Anforderungen zum Festlegen von Zugriffssteuerungslisten (ACLs) oder zum Aktualisieren von ACLs fehl und geben den Fehlercode `AccessControlListNotSupported` zurück. Anfragen zum Lesen von ACLs werden weiterhin unterstützt.

Um ein Objekt über eine ACL öffentlich lesbar zu machen, erteilen Sie der Gruppe `AllUsers` die READ-Berechtigung, wie im folgenden Berechtigungselement dargestellt. Fügen Sie dieses Rechteelement der Objekt-ACL hinzu. Weitere Informationen zur Verwaltung von ACLs finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht.

```
<Grant>
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:type="Group">
    <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
  </Grantee>
  <Permission>READ</Permission>
</Grant>
```

(Optional) Protokollieren des Webdatenverkehrs

Sie können optional die Amazon-S3-Server-Zugriffsprotokollierung für einen Bucket aktivieren, der als statische Website konfiguriert ist. Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anfragen, die an Ihren Bucket gestellt werden. Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung. Wenn Sie Amazon verwenden möchten, CloudFront um Ihre Website zu beschleunigen, können Sie auch die - CloudFront Protokollierung verwenden. Weitere Informationen finden Sie unter Konfigurieren und Verwenden von Zugriffsprotokollen im Amazon- CloudFront Entwicklerhandbuch.

So aktivieren Sie die Server-Zugriffsprotokollierung für Ihren statischen Website-Bucket

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Erstellen Sie in derselben Region, in der Sie den Bucket erstellt haben, der als statische Website konfiguriert ist, einen Bucket für die Protokollierung, z. B. `logs.example.com`.
3. Erstellen Sie einen Ordner für die Protokolldateien der Server-Zugriffsprotokollierung (z. B, `logs`).
4. (Optional) Wenn Sie verwenden möchten, um die Leistung Ihrer Website CloudFront zu verbessern, erstellen Sie einen Ordner für die CloudFront Protokolldateien (z. B. `cdn`).

   Weitere Informationen finden Sie unter Beschleunigen Ihrer Website mit Amazon CloudFront.
5. Wählen Sie in der Liste Buckets den Namen Ihres Buckets.
6. Wählen Sie Properties (Eigenschaften).
7. Wählen Sie unter Server access logging (Server-Zugriffsprotokollierung) Edit (Bearbeiten).
8. Wählen Sie Enable aus.
9. Wählen Sie im Ziel-Bucket den Bucket und das Ordnerziel für die Server-Zugriffsprotokolle aus:
   - Navigieren Sie zum Ordner- und Bucket-Speicherort:
     1. Wählen Sie Browse S3 (S3 durchsuchen).
     2. Wählen Sie den Bucket-Namen und dann den Ordner mit den Protokollen aus.
     3. Wählen Sie Choose path (Pfad wählen).
   - Geben Sie den S3-Bucket-Pfad ein, z. B, **`s3://logs.example.com/logs/`**.
10. Wählen Sie Save Changes (Änderungen speichern).

    In Ihrem Protokoll-Bucket können Sie jetzt auf Ihre Protokolle zugreifen. Amazon S3 schreibt Website-Zugriffsprotokolle alle zwei Stunden in Ihren Bucket zur Protokollierung.

(Optional) Konfigurieren einer Webseitenumleitung

Wenn Ihr Amazon-S3-Bucket für das statische Website-Hosting konfiguriert ist, können Sie Umleitungsregeln für Ihren Bucket oder die darin enthaltenen Objekte konfigurieren. Sie haben die folgenden Optionen für die Konfigurierung einer Umleitung.

Themen

- Umleiten von Anforderungen für den Website-Endpunkt Ihres Buckets an einen anderen Bucket oder eine andere Domäne
- Konfigurieren von Umleitungsregeln für die Verwendung von fortschrittliche bedingten Umleitungen
- So leiten Sie Anforderungen für ein Objekt um

Umleiten von Anforderungen für den Website-Endpunkt Ihres Buckets an einen anderen Bucket oder eine andere Domäne

Sie können alle Anforderungen für einen Website-Endpunkt eines Buckets an einen anderen Host umleiten. Wenn Sie alle Anforderungen umleiten, werden alle Anforderungen an den Website-Endpunkt an den angegebenen Bucket oder die Domäne umgeleitet.

Wenn Ihre Root-Domäne zum Beispiel `example.com` ist, und Sie Anfragen sowohl für `http://example.com`, als auch für `http://www.example.com` bereitstellen möchten, müssen Sie zwei Buckets mit den Namen `example.com` und `www.example.com` erstellen. Pflegen Sie dann den Inhalt im Bucket `example.com` und konfigurieren Sie den anderen Bucket `www.example.com` so, dass alle Anforderungen an den Bucket `example.com` umgeleitet werden. Weitere Informationen finden Sie unter Konfigurieren einer statischen Website mit einem benutzerdefinierten Domänennamen.

So leiten Sie Anforderungen für einen Bucket-Website-Endpunkt um

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie unter Buckets den Namen des Buckets aus, von dem Sie Anfragen umleiten möchten (z. B. `www.example.com`).
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Static website hosting (Hosting statischer Websites) Edit (Bearbeiten) aus.
5. Wählen Sie Redirect requests for an object (Anfragen für ein Objekt umleiten).
6. Geben Sie im Feld Host name (Host-Name) den Website-Endpunkt für Ihren Bucket oder Ihre benutzerdefinierte Domäne ein.

   Wenn Sie Anforderungen beispielsweise zu einer Root-Domänenadresse umleiten, geben Sie ei **`example.com`**.
7. Wählen Sie unter Protocol (Protokoll) das Protokoll für die umgeleiteten Anfragen (none (kein), http oder https).

Wenn Sie kein Protokoll angeben, ist die Standardoption none (kein).

8. Wählen Sie Save Changes (Änderungen speichern).

Konfigurieren von Umleitungsregeln für die Verwendung von fortschrittliche bedingten Umleitungen

Unter Verwendung fortschrittliche Umleitungsregeln können Sie bedingt Anfragen abhängig von bestimmten Objektschlüsselnamen oder Präfixen in der Anfrage weiterleiten, oder abhängig von Antwortcodes. Angenommen, Sie löschen ein Objekt in Ihrem Bucket oder benennen es um. Sie können eine Weiterleitungsregel hinzufügen, die die Anfrage an ein anderes Objekt weiterleitet. Wenn Sie einen Ordner nicht zur Verfügung stellen wollen, können Sie eine Umleitungsregel hinzufügen, um die Anfrage an eine andere Webseite umzuleiten. Sie können auch eine Weiterleitungsregel hinzufügen, um Fehlerbedingungen zu verarbeiten, indem Sie Anfragen, die den Fehler zurückgeben, an eine andere Domäne weiterleiten, wenn der Fehler verarbeitet wird.

Wenn Sie das statische Website-Hosting für Ihren Bucket aktivieren, können Sie optional fortschrittliche Umleitungsregeln angeben. Amazon S3 hat eine Begrenzung von 50 Routingregeln pro Websitekonfiguration. Wenn Sie mehr als 50 Routingregeln benötigen, können Sie die Objektumleitung verwenden. Weitere Informationen finden Sie unter Verwenden der S3-Konsole.

Weitere Informationen zum Konfigurieren von Routingregeln mit der REST-API finden Sie unter PutBucketWebsite in der API-Referenz zu Amazon Simple Storage Service.

> **Important**
>
> Um Umleitungsregeln in der neuen Amazon-S3-Konsole zu erstellen, müssen Sie JSON verwenden. JSON-Beispiele finden Sie unter Beispiele für Umleitungsregeln.

So konfigurieren Sie Umleitungsregeln für eine statische Website

Gehen Sie folgendermaßen vor, um Umleitungsregeln für einen Bucket hinzuzufügen, für den das statische Website-Hosting bereits aktiviert ist.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets (Buckets) den Namen eines Buckets aus, den Sie als statische Website konfiguriert haben.

3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie unter Static website hosting (Hosting statischer Websites) Edit (Bearbeiten) aus.
5. Geben Sie im Feld Redirection rules (Umleitungsregeln) Ihre Umleitungsregeln in JSON ein.

   In der S3-Konsole beschreiben Sie die Regeln mit JSON. JSON-Beispiele finden Sie unter Beispiele für Umleitungsregeln. Amazon S3 hat eine Begrenzung von 50 Routingregeln pro Websitekonfiguration.
6. Wählen Sie Save Changes (Änderungen speichern).

Routingregel-Elemente

Das Folgende ist eine allgemeine Syntax zum Definieren der Routingregeln in einer Website-Konfiguration in JSON und XML. Zum Konfigurieren von Umleitungsregeln in der neuen S3-Konsole müssen Sie JSON verwenden. JSON-Beispiele finden Sie unter Beispiele für Umleitungsregeln.

JSON

```
[
    {
      "Condition": {
        "HttpErrorCodeReturnedEquals": "string",
        "KeyPrefixEquals": "string"
      },
      "Redirect": {
        "HostName": "string",
        "HttpRedirectCode": "string",
        "Protocol": "http"|"https",
        "ReplaceKeyPrefixWith": "string",
        "ReplaceKeyWith": "string"
      }
    }
  ]

Note: Redirect must each have at least one child element. You can have either
 ReplaceKeyPrefix with or ReplaceKeyWith but not both.
```

XML

```
<RoutingRules> =
    <RoutingRules>
```

```
          <RoutingRule>...</RoutingRule>
          [<RoutingRule>...</RoutingRule>
          ...]
     </RoutingRules>

<RoutingRule> =
   <RoutingRule>
      [ <Condition>...</Condition> ]
      <Redirect>...</Redirect>
   </RoutingRule>

<Condition> =
   <Condition>
      [ <KeyPrefixEquals>...</KeyPrefixEquals> ]
      [ <HttpErrorCodeReturnedEquals>...</HttpErrorCodeReturnedEquals> ]
   </Condition>
    Note: <Condition> must have at least one child element.

<Redirect> =
   <Redirect>
      [ <HostName>...</HostName> ]
      [ <Protocol>...</Protocol> ]
      [ <ReplaceKeyPrefixWith>...</ReplaceKeyPrefixWith>  ]
      [ <ReplaceKeyWith>...</ReplaceKeyWith> ]
      [ <HttpRedirectCode>...</HttpRedirectCode> ]
   </Redirect>

Note: <Redirect> must have at least one child element. You can have either
 ReplaceKeyPrefix with or ReplaceKeyWith but not both.
```

In der folgenden Tabelle werden die Elemente in der Weiterleitungsregel beschrieben.

Name	Beschreibung
`RoutingRules`	Container für eine Sammlung von `RoutingRule` -Elementen.
`RoutingRule`	Eine Regel, die eine Bedingung identifiziert, sowie die Weiterlei tung, die angewendet wird, wenn die Bedingung erfüllt ist. Bedingung:

Name	Beschreibung
	• Ein `RoutingRules` -Container muss mindestens eine Weiterleitungsregel enthalten.
`Condition`	Container für die Beschreibung einer Bedingung, die für die angegebene Weiterleitung erfüllt sein muss, damit sie angewende t wird. Wenn die Weiterleitungsregel keine Bedingung enthält, wird die Regel auf alle Anfragen angewendet.
`KeyPrefixEquals`	Das Präfix des Objektschlüsselnamens, von dem die Anfragen weitergeleitet werden. `KeyPrefixEquals` ist erforderliche, wenn `HttpError CodeReturnedEquals` nicht angegeben ist. Wenn `KeyPrefixEquals` und `HttpErrorCodeReturnedEquals` angegeben sind, müssen beide true sein, damit die Bedingung erfüllt ist.
`HttpErrorCodeRetur nedEquals`	Der HTTP-Fehlercode, mit dem eine Übereinstimmung vorliegen muss, damit die Umleitung angewendet wird. Wenn ein Fehler auftritt und der Fehlercode mit diesem Wert übereinstimmt, gilt die angegebene Weiterleitung. `HttpErrorCodeReturnedEquals` ist erforderliche, wenn `KeyPrefixEquals` nicht angegeben ist. Wenn `KeyPrefix Equals` und `HttpErrorCodeReturnedEquals` angegeben sind, müssen beide true sein, damit die Bedingung erfüllt ist.

Name	Beschreibung
`Redirect`	Container-Element, das Anweisungen für die Weiterleitung der Anfrage enthält. Sie können Anfragen an einen anderen Host oder eine andere Seite umleiten, oder ein anderes Protokoll vorgeben. Eine `RoutingRule` muss ein `Redirect`-Element besitzen. Ein `Redirect`-Element muss mindestens eines der folgenden zugeordneten Elemente enthalten: `Protocol`, `HostName`, `ReplaceKeyPrefixWith` , `ReplaceKeyWith` oder `HttpRedirectCode` .
`Protocol`	Das Protokoll, `http` oder `https`, das im `Location`-Header verwendet werden soll, der in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `Protocol` nicht erforderlich.
`HostName`	Der Hostname, der im `Location`-Header verwendet werden soll, der in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `HostName` nicht erforderlich.
`ReplaceKeyPrefixWith`	Das Präfix des Objektschlüsselnamens, der den Wert von `KeyPrefixEquals` in der Umleitungsanforderung ersetzt. Wenn eines der zugeordneten Elemente vorhanden ist, ist `ReplaceKeyPrefixWith` nicht erforderlich. Es kann nur bereitgestellt werden, wenn `ReplaceKeyWith` nicht bereitgestellt wird.

Name	Beschreibung
`ReplaceKeyWith`	Der Objektschlüssel, der im `Location`-Header verwendet werden soll und in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `ReplaceKeyWith` nicht erforderlich. Es kann nur bereitgestellt werden, wenn `ReplaceKeyPrefixWith` nicht bereitgestellt wird.
`HttpRedirectCode`	Der HTTP-Umleitungscode, der im `Location`-Header verwendet werden soll, der in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `HttpRedirectCode` nicht erforderlich.

Beispiele für Umleitungsregeln

Die folgenden Beispiele erklären häufige Weiterleitungsfälle:

> **Important**
>
> Um Umleitungsregeln in der neuen Amazon-S3-Konsole zu erstellen, müssen Sie JSON verwenden.

Example 1: Weiterleitung nach der Umbenennung eines Schlüsselpräfix

Angenommen, Ihr Bucket enthält die folgenden Objekte:

- index.html
- docs/article1.html
- docs/article2.html

Jetzt wollen Sie den Ordner von `docs/` in `documents/` umbenennen. Nach dieser Änderung müssen Sie Anfragen für das Präfix `docs/` in `documents/` weiterleiten. Beispielsweise muss die Anfragen für `docs/article1.html` an `documents/article1.html` weitergeleitet werden.

In diesem Fall fügen Sie der Konfiguration der Website die folgende Routingregel hinzu.

JSON

```
[
    {
        "Condition": {
            "KeyPrefixEquals": "docs/"
        },
        "Redirect": {
            "ReplaceKeyPrefixWith": "documents/"
        }
    }
]
```

XML

```
  <RoutingRules>
    <RoutingRule>
    <Condition>
      <KeyPrefixEquals>docs/</KeyPrefixEquals>
    </Condition>
    <Redirect>
      <ReplaceKeyPrefixWith>documents/</ReplaceKeyPrefixWith>
    </Redirect>
    </RoutingRule>
  </RoutingRules>
```

Example 2: Weiterleitung von Anfragen für einen gelöschten Ordner auf eine Seite

Angenommen, Sie löschen den Ordner `images/` (d. h. Sie löschen alle Objekte mit dem Schlüsselpräfix `images/`). Sie können eine Weiterleitungsregel einrichten, die Anfrage für jedes Objekt mit dem Schlüsselpräfix `images/` auf eine Seite namens `folderdeleted.html` weiterleitet.

JSON

```
[
```

```
    {
        "Condition": {
            "KeyPrefixEquals": "images/"
        },
        "Redirect": {
            "ReplaceKeyWith": "folderdeleted.html"
        }
    }
]
```

XML

```
  <RoutingRules>
    <RoutingRule>
    <Condition>
       <KeyPrefixEquals>images/</KeyPrefixEquals>
    </Condition>
    <Redirect>
      <ReplaceKeyWith>folderdeleted.html</ReplaceKeyWith>
    </Redirect>
    </RoutingRule>
  </RoutingRules>
```

Example 3: Weiterleitung für einen HTTP-Fehler.

Angenommen, Sie möchten Anforderungen an eine Amazon Elastic Compute Cloud (Amazon EC2)-Instance umleiten, wenn ein angefordertes Objekt nicht gefunden wird. Fügen Sie eine Umleitungsregel hinzu, sodass bei einem HTTP-Statuscode 404 (Not Found) der Besucher der Site an eine Amazon EC2-Instance weitergeleitet wird, die die Anforderung verarbeitet.

Im folgenden Beispiel wird auch das Objektschlüsselpräfix `report-404/` in die Weiterleitung eingefügt. Wenn Sie beispielsweise eine Seite mit der Bezeichnung `ExamplePage.html` anfordern und ein HTTP-404-Fehler ausgegeben wird, wird die Anforderung an eine Seite mit dem Namen `report-404/ExamplePage.html` auf der angegebenen Amazon EC2-Instance weitergeleitet. Wenn es keine Weiterleitungsregel gibt und der HTTP-Fehler 404 auftritt, wird das in der Konfiguration festgelegte Fehlerdokument zurückgegeben.

JSON

```
[
```

```
    {
        "Condition": {
            "HttpErrorCodeReturnedEquals": "404"
        },
        "Redirect": {
            "HostName": "ec2-11-22-333-44.compute-1.amazonaws.com",
            "ReplaceKeyPrefixWith": "report-404/"
        }
    }
]
```

XML

```
  <RoutingRules>
    <RoutingRule>
    <Condition>
      <HttpErrorCodeReturnedEquals>404</HttpErrorCodeReturnedEquals >
    </Condition>
    <Redirect>
      <HostName>ec2-11-22-333-44.compute-1.amazonaws.com</HostName>
      <ReplaceKeyPrefixWith>report-404/</ReplaceKeyPrefixWith>
    </Redirect>
    </RoutingRule>
  </RoutingRules>
```

So leiten Sie Anforderungen für ein Objekt um

Sie können Anforderungen für ein Objekt an ein anderes Objekt oder eine andere URL umleiten, indem Sie in den Metadaten des Objekts den Speicherort für die Websiteumleitung festlegen. Sie richten die Umleitung ein, indem Sie den Objekt-Metadaten die `x-amz-website-redirect-location`-Eigenschaft hinzufügen. In der Amazon-S3-Konsole legen Sie den Wert für Website Redirect Location (Websiteumleitungsort) in den Metadaten des Objekts fest. Wenn Sie die Amazon-S3-API verwenden, stellen Sie `x-amz-website-redirect-location` ein. Die Website interpretiert anschließend das Objekt als eine 301-Umleitung.

Um eine Anfrage an ein anderes Objekt umzuleiten, richten Sie den Umleitungsstandort auf den Schlüssel für das Zielobjekt ein. Um eine Anfrage an eine externe URL umzuleiten, richten Sie den Umleitungsstandort auf die gewünschte URL ein. Weitere Informationen zu Objekt-Metadaten erhalten Sie unter Systemdefinierte Objektmetadaten.

Wenn Sie eine Seitenumleitung einrichten, können Sie den Inhalt des Quellobjekts beibehalten oder löschen. Wenn es in Ihrem Bucket beispielsweise das Objekt `page1.html` gibt, können Sie alle Anforderungen für diese Seite an ein anderes Objekt umleiten, `page2.html`. Sie haben hierfür zwei Möglichkeiten:

- Behalten Sie den Inhalt des Objekts `page1.html` und leiten Sie Seitenanforderungen um.
- Löschen Sie den Inhalt von `page1.html` und laden Sie ein Null-Byte-Objekt mit dem Namen `page1.html` hoch, um das vorhandene Objekt zu ersetzen und Seitenanforderungen umzuleiten.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie als statische Website konfiguriert haben (z. B. `example.com`).
3. Wählen Sie unter Objects (Objekte) Ihr Objekt aus.
4. Wählen Sie Actions (Aktionen) und dann Edit metadata (Metadaten bearbeiten).
5. Wählen Sie Metadata (Metadaten) aus.
6. Wählen Sie Add Metadata (Metadaten hinzufügen) aus.
7. Wählen Sie unter Type (Typ) die Option System Defined (Systemdefiniert) aus.
8. Wählen Sie unter Schlüssel die Option x-amz-website-redirect-location aus.
9. Geben Sie in Value (Wert) den Schlüsselnamen des Objekts ein, zu dem Sie umleiten möchten, z. B. `/page2.html`.

   Für ein anderes Objekt im selben Bucket ist das Präfix `/` im Wert erforderlich. Sie können den Wert auch auf eine externe URL festlegen, z. B, `http://www.example.com`.
10. Wählen Sie Edit metadata (Metadaten bearbeiten).

Verwenden der REST-API

Die folgenden Amazon-S3-API-Aktionen unterstützen den `x-amz-website-redirect-location`-Header in der Anforderung. Amazon S3 speichert den Header-Wert in den Objekt-Metadaten als `x-amz-website-redirect-location`.

- PUT Object
- Initiieren eines mehrteiligen Uploads

- POST Object
- PUT Object – Kopieren

Ein Bucket, der für das Website-Hosting konfiguriert ist, hat sowohl den Website-Endpunkt als auch den REST-Endpunkt. Eine Anfrage für eine Seite, die als 301-Umleitung konfiguriert ist, erzeugt die folgenden möglichen Ergebnisse, abhängig vom Endpunkt der Anfrage:

- Regionsspezifischer Website-Endpunkt – Amazon S3 leitet die Seitenanforderung in Übereinstimmung mit dem Wert der `x-amz-website-redirect-location`-Eigenschaft um.
- REST-Endpunkt – Amazon S3 leitet die Seitenanforderung nicht um Es gibt das angefragte Objekt zurück.

Weitere Informationen zu den Endpunkten finden Sie unter Wichtige Unterschiede zwischen einem Website-Endpunkt und einem REST-API-Endpunkt.

Wenn Sie eine Seitenumleitung einrichten, können Sie den Inhalt des Objekts beibehalten oder löschen. Nehmen wir z. B. an, dass Sie ein `page1.html`-Objekt in Ihrem Bucket haben.

- Um den Inhalt von `page1.html` beizubehalten und nur Seitenanforderungen umzuleiten, können Sie eine PUT Object - Copy-Anforderung stellen, um ein neues `page1.html`-Objekt zu erstellen, das das vorhandene `page1.html`-Objekt als Quelle verwendet. In Ihrer Anfrage richten Sie den `x-amz-website-redirect-location`-Header ein. Nachdem die Anforderung abgeschlossen ist, ist die Originalseite mit ihrem Inhalt unverändert, aber Amazon S3 leitet alle Anforderungen für die Seite an den von Ihnen angegebenen Umleitungsstandort um.
- Um den Inhalt des `page1.html`-Objekts zu löschen und Anforderungen für die Seite umzuleiten, können Sie eine PUT-Objektanforderung senden, um ein Nullbyte-Objekt hochzuladen, das den gleichen Objektschlüssel hat: `page1.html`. In der PUT-Anfrage setzen Sie `x-amz-website-redirect-location` für `page1.html` auf das neue Objekt. Nachdem die Anfrage abgeschlossen ist, hat `page1.html` keinen Inhalt mehr, und Anfragen werden an den Standort umgeleitet, der in `x-amz-website-redirect-location` angegeben ist.

Wenn Sie das Objekt mit der Aktion GET Object abrufen, gibt Amazon S3 zusammen mit anderen Objekt-Metadaten den `x-amz-website-redirect-location`-Header in der Antwort zurück.

Entwickeln mit Amazon S3

Dieser Abschnitt enthält entwicklerbezogene Themen zur Verwendung von Amazon S3. Weitere Informationen finden Sie in den nachstehenden Themen.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Senden von Anforderungen

Amazon S3 ist ein REST-Service. Sie können Anfragen an Amazon S3 über die REST-API oder die Wrapper-Bibliotheken des AWS-SDK senden (siehe Beispiel-Code und Bibliotheken), die die zugrunde liegende Amazon-S3-REST-API umschließen. Dies vereinfacht Ihre Programmier-Aufgaben.

Jede Interaktion mit Amazon S3 erfolgt entweder authentifiziert oder anonym. Die Authentifizierung ist ein Vorgang, bei dem die Identität des Auftraggebers, der auf ein Amazon Web Services (AWS) Produkt zugreifen möchte, überprüft wird. Authentifizierte Anforderungn müssen einen Signaturwert enthalten, der den Sender der Anforderung authentifiziert. Ein Teil des Signaturwerts wird von den AWS-Zugriffsschlüsseln des Auftraggebers generiert (Zugriffsschlüssel-ID und geheimer Zugriffsschlüssel). Weitere Informationen zum Abrufen von Zugriffsschlüsseln finden Sie unter Wie erhalte ich Sicherheitsanmeldeinformationen? in der Allgemeine AWS-Referenz.

Wenn Sie das AWS-SDK verwenden, berechnen die Bibliotheken die Signatur anhand der von Ihnen bereitgestellten Schlüssel. Wenn Sie in Ihrer Anwendung jedoch direkte REST-API-Aufrufe senden, müssen Sie den Code für die Berechnung der Signatur schreiben und diesen der Anforderung hinzufügen.

Themen

Über Zugriffsschlüssel

In den folgenden Abschnitten werden die Arten der Zugriffsschlüssel vorgestellt, die Sie verwenden können, um authentifizierte Anfragen zu senden.

AWS-Konto-Zugriffsschlüssel

Die Kontozugriffsschlüssel stellen vollständigen Zugriff auf die AWS-Ressourcen bereit, die das jeweilige Konto besitzt. Im Folgenden finden Sie Beispiele für Zugriffsschlüssel:

- Zugriffsschlüssel-ID (eine alphanumerische Zeichenfolge mit 20 Zeichen). Zum Beispiel: AKIAIOSFODNN7EXAMPLE
- Geheimer Zugriffsschlüssel (eine Zeichenfolge mit 40 Zeichen). Zum Beispiel: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

Die Zugriffsschlüssel-ID identifiziert ein AWS-Konto auf spezifische Weise. Sie können diese Zugriffsschlüssel verwenden, um authentifizierte Anforderungen an Amazon S3 zu senden.

IAM-Benutzer-Zugriffsschlüssel

Sie können ein einzelnes AWS-Konto für Ihr Unternehmen erstellen. Möglicherweise gibt es jedoch mehrere Mitarbeiter in der Organisation, die auf die AWS-Ressourcen Ihrer Organisation zugreifen müssen. Wenn Sie die Zugriffsschlüssel für Ihr AWS-Konto freigeben, wird die Sicherheit reduziert. Das Erstellen einzelner AWS-Konten für jeden Mitarbeiter ist jedoch vielleicht nicht praktikabel.

Darüber hinaus können Sie Ressourcen wie Buckets und Objekte nicht einfach freigeben, da sie im Besitz verschiedener Konten sind. Um Ressourcen freizugeben, müssen Sie Berechtigungen gewähren. Dies bedeutet zusätzlichen Aufwand.

In solchen Szenarien können Sie AWS Identity and Access Management (IAM) zum Erstellen von Benutzern mit eigenen Zugriffsschlüsseln in Ihrem AWS-Konto verwenden und IAM-Benutzerrichtlinien anfügen, die diesen Benutzern die entsprechenden Zugriffsberechtigungen für Ressourcen gewähren. Um diese Benutzer besser verwalten zu können, ermöglicht IAM Ihnen das Erstellen von Benutzergruppen und das Gewähren von Berechtigungen auf Gruppenebene, die für alle Benutzer in der betreffenden Gruppe gelten.

Diese Benutzer werden als IAM-Benutzer bezeichnet. Sie erstellen und verwalten diese innerhalb von AWS. Das übergeordnete Konto steuert die Möglichkeiten der Benutzer, auf zuzugreife AWS. Alle von einem IAM-Benutzer erstellten Ressourcen werden über das übergeordnete AWS-Konto gesteuert und bezahlt. Diese IAM-Benutzer können mittels ihrer eigenen Sicherheitsanmeldeinformationen authentifizierte Anfragen an Amazon S3 senden. Weitere Informationen zum Erstellen und Verwalten von Benutzern in Ihrem AWS-Konto finden Sie auf der Seite mit AWS Identity and Access Management-Produktdetails.

Temporäre Sicherheitsanmeldeinformationen

IAM ermöglicht Ihnen nicht nur das Erstellen von IAM-Benutzern mit eigenen Zugriffsschlüsseln, sondern auch die Gewährung temporärer Sicherheitsanmeldeinformationen (temporärer Zugriffsschlüssel und eines Sicherheitstokens) für IAM-Benutzer, damit diese auf Ihre AWS-Services und -Ressourcen zugreifen können. Sie können Benutzer auch außerhalb von in Ihrem System verwalte AWS. Diese Benutzer werden als verbundene Benutzer bezeichnet. Zusätzlich kann es sich bei Benutzern um Anwendungen handeln, die Sie erstellen, um auf Ihre AWS-Ressourcen zuzugreifen.

IAM stellt die AWS Security Token Service-API bereit, über die Sie temporäre Sicherheitsanmeldeinformationen anfordern können. Sie können entweder die AWS-STS-API oder das AWS-SDK verwenden, um diese Anmeldeinformationen anzufordern. Die API gibt temporäre Sicherheitsanmeldeinformationen (Zugriffsschlüssel-ID und geheimen Zugriffsschlüssel) und ein Sicherheitstoken zurück. Diese Anmeldeinformationen sind nur für die Dauer gültig, die Sie bei der Anforderung angeben. Sie verwenden die Zugriffsschlüssel-ID und den geheimen Schlüssel auf dieselbe Weise, wie Sie diese beim Senden von Anfragen über Ihr AWS-Konto oder IAM-Benutzerzugriffsschlüssel verwenden. Zusätzlich muss jede Anfrage, die Sie an Amazon S3 senden, das Token enthalten.

Ein IAM-Benutzer kann diese temporären Sicherheitsanmeldeinformationen für die eigene Verwendung anfordern oder sie an verbundene Benutzer oder Anwendungen vergeben. Wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, in der die Berechtigungen definiert sind, die Sie mit diesen temporären Sicherheitsanmeldeinformationen verknüpfen möchten. Ein verbundener Benutzer kann nicht mehr Berechtigungen als der übergeordnete IAM-Benutzer erhalten, der die temporären Anmeldeinformationen angefordert hat.

Sie können diese temporären Sicherheitsanmeldeinformationen beim Senden von Anfragen an Amazon S3 verwenden. Die API-Bibliotheken berechnen anhand dieser Anmeldeinformationen den notwendigen Signaturwert, um Ihre Anforderung zu authentifizieren. Wenn Sie beim Senden von Anfragen abgelaufene Anmeldeinformationen verwenden, lehnt Amazon S3 die Anfrage ab.

Informationen zum Signieren von Anfragen mittels temporärer Sicherheitsanmeldeinformationen in Ihren REST-API-Anfragen finden Sie unter Signieren und Authentifizieren von REST-Anforderungen. Informationen zum Senden von Anfragen mittels AWS-SDKs finden Sie unter Senden von Anfragen unter Verwendung der AWS-SDKs.

Weitere Informationen zur Unterstützung temporärer Anmeldeinformationen in IAM finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch.

Um die Sicherheit zu verbessern, können Sie für den Zugriff auf Ihre Amazon-S3-Ressourcen eine Multi-Factor Authentication (MFA) anfordern, indem Sie eine Bucket-Richtlinie konfigurieren. Weitere Informationen finden Sie unter Verlangen von MFA. Nachdem Sie für den Zugriff auf Ihre Amazon-S3-Ressourcen eine MFA angefordert haben, können Sie auf diese Ressourcen nur durch die Bereitstellung temporärer Anmeldeinformationen zugreifen, die mittels eines MFA-Schlüssels erstellt wurden. Weitere Informationen finden Sie auf der Seite mit Details zur AWS-Multi-Factor Authentication und unter Configuring MFA-Protected API Access (Konfigurieren eines MFA-geschützten API-Zugriffs) im IAM-Benutzerhandbuch.

Anforderungsendpunkte

Sie senden REST-Anfragen an den vordefinierten Endpunkt des Service. Die Liste aller AWS-Services und ihrer entsprechenden Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.

Stellen von Anforderungen an Amazon S3 über IPv6

Amazon Simple Storage Service (Amazon S3) unterstützt zusätzlich zum IPv4-Protokoll die Möglichkeit, mit dem Internet Protocol Version 6 (IPv6) auf S3-Buckets zuzugreifen. Amazon-S3-Dual-Stack-Endpunkte unterstützen Anforderungen an S3-Buckets über IPv6 und IPv4. Für den Zugriff auf Amazon S3 über IPv6 fallen keine zusätzlichen Gebühren an. Weitere Informationen zu Preisen finden Sie unter Amazon-S3-Preise.

Themen

Erste Schritte für Anforderungen über IPv6

Um eine Anforderung für einen S3-Bucket über IPv6 zu erstellen, brauchen Sie einen Dual-Stack-Endpunkt. Der nächste Abschnitt beschreibt Anfragen über IPv6 unter Verwendung von Dual-Stack-Endpunkten.

Nachfolgend sind einige Dinge beschrieben, die Sie wissen sollten, bevor Sie versuchen, über IPv6 auf einen Bucket zuzugreifen.

- Der Client und das Netzwerk, die auf den Bucket zugreifen, müssen für IPv6 aktiviert sein.
- Für den IPv6-Zugriff werden Anforderungen im virtuellen Hosting- und im Pfad-Stil unterstützt. Weitere Informationen finden Sie unter Amazon-S3-Dual-Stack-Endpunkte.
- Wenn Sie eine IP-Quelladressen-Filterung in Ihren AWS Identity and Access Management (IAM) Benutzer- oder -Bucket-Richtlinien verwenden, müssen Sie die Richtlinien aktualisieren, um IPv6-Adressbereiche zu berücksichtigen. Weitere Informationen finden Sie unter Verwenden von IPv6-Adressen in IAM-Richtlinien.
- Bei Verwendung von IPv6 geben die Serverzugriff-Protokolldateien IP-Adressen in einem IPv6-Format aus. Sie müssen vorhandene Tools, Skripts und Software aktualisieren, mit denen Sie Amazon-S3-Protokolldateien analysieren, sodass sie die mit IPv6 formatierte `Remote IP`-Adressen analysieren können. Weitere Informationen erhalten Sie unter Amazon-S3-Server-Zugriffsprotokollformat und Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

> **Note**
>
> Wenn Sie Probleme mit IPv6-Adressen in Protokolldateien haben, wenden Sie sich an den AWS Support.

Anforderungen über IPv6 unter Verwendung von Dual-Stack-Endpunkten

Sie stellen Anforderungen Amazon-S3-API-Aufrufen über IPv6, indem Sie Dual-Stack-Endpunkte verwenden. Die Amazon-S3-API-Vorgänge funktionieren genauso, egal ob Sie über IPv6 oder über IPv4 auf Amazon S3 zugreifen. Die Leistung sollte ebenfalls dieselbe bleiben.

Wenn Sie die REST-API verwenden, greifen Sie direkt auf einen Dual-Stack-Endpunkt zu. Weitere Informationen finden Sie unter Dual-Stack-Endpunkte.

Wenn Sie die AWS Command Line Interface (AWS CLI) und AWS SDKs verwenden, können Sie einen Parameter oder ein Flag verwenden, um zu einem Dual-Stack-Endpunkt zu wechseln. Sie können den Dual-Stack-Endpunkt auch direkt als Override des Amazon-S3-Endpunkts in der Konfigurationsdatei angeben.

Sie können einen Dual-Stack-Endpunkt verwenden, um über IPv6 auf einen Bucket zuzugreifen. Dazu können Sie Folgendes verwenden:

- Die AWS CLI; siehe Verwenden von Dual-Stack-Endpunkten von der AWS CLI.
- Die AWS-SDKs finden Sie unter Dual-Stack-Endpunkte von der AWS-SDKs verwenden.
- Die REST-API, siehe Senden von Anforderungen an Dual-Stack-Endpunkte unter Verwendung der REST-API.

Funktionen, die über IPv6 nicht zur Verfügung stehen

Die folgende Funktion wird derzeit beim Zugriff auf einen S3-Bucket über IPv6 nicht unterstützt: Statisches Website-Hosting von einem S3-Bucket aus.

Verwenden von IPv6-Adressen in IAM-Richtlinien

Bevor Sie versuchen, mit IPv6 auf einen Bucket zuzugreifen, müssen Sie sicherstellen, dass alle IAM-Benutzer- oder S3-Bucket-Richtlinien, die für die IP-Adressfilterung verwendet werden,

aktualisiert werden, um den IPv6-Adressbereich zu berücksichtigen. Richtlinien für die IP-Adressfilterung, die nicht für die Verarbeitung von IPv6-Adressen aktualisiert werden, führen womöglich dazu, dass Clients den Zugriff auf den Bucket fälschlicherweise verlieren oder erhalten, wenn sie beginnen, IPv6 zu verwenden. Weitere Informationen über die Verwaltung von Zugriffsberechtigungen mit IAM finden Sie unter Identity and Access Management in Amazon S3.

IAM-Richtlinien, die IP-Adressen filtern, verwenden Bedingungsoperatoren für IP-Adressen. Die folgende Bucket-Richtlinie identifiziert den Bereich 54.240.143.* als Bereich zulässiger IPv4-Adressen durch Verwendung von Bedingungsoperatoren für IP-Adressen. Alle IP-Adressen außerhalb dieses Bereichs erhalten keinen Zugriff auf den Bucket (`examplebucket`). Alle IPv6-Adressen liegen außerhalb des zulässigen Bereichs, deshalb verhindert diese Richtlinie, dass IPv6-Adressen auf zugreife `examplebucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::examplebucket/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
      }
    }
  ]
}
```

Sie können das `Condition`-Element der Bucket-Richtlinie ändern, um die IPv4- (`54.240.143.0/24`) und IPv6- (`2001:DB8:1234:5678::/64`) Adressbereiche zuzulassen, wie im folgenden Beispiel gezeigt. Sie können denselben Typ `Condition`-Block verwenden, wie im Beispiel gezeigt, um Ihre IAM-Benutzer- und Bucket-Richtlinien zu aktualisieren.

```
       "Condition": {
         "IpAddress": {
            "aws:SourceIp": [
              "54.240.143.0/24",
               "2001:DB8:1234:5678::/64"
             ]
          }
```

```
        }
```

Bevor Sie IPv6 verwenden, müssen Sie alle relevanten IAM-Benutzer- und S3-Bucket-Richtlinien aktualisieren, die eine IP-Adressfilterung verwenden, um die IPv6-Adressbereiche zu berücksichtigen. Wir empfehlen Ihnen, Ihre IAM-Richtlinien mit den IPv6-Adressbereichen Ihres Unternehmens zu aktualisieren, ebenso wie mit Ihren vorhandenen IPv4-Adressbereichen. Ein Beispiel für eine Bucket-Richtlinie, die den Zugriff über IPv6 und IPv4 gestattet, finden Sie unter Beschränken des Zugriffs auf bestimmte IP-Adressen.

Sie können Ihre IAM-Benutzerrichtlinien mit der IAM-Konsole unter https://console.aws.amazon.com/iam/ überprüfen. Weitere Informationen zu IAM finden Sie im IAM-Benutzerhandbuch. Weitere Informationen zum Bearbeiten von S3-Bucket-Richtlinien finden Sie unter Hinzufügen einer Bucket-Richtlinie mit der Amazon-S3-Konsole.

Testen der IP-Adresskompatibilität

Wenn Sie Linux/Unix oder Mac OS X verwenden, können sie testen, ob Sie über IPv6 auf einen Dual-Stack-Endpunkt zugreifen können, indem Sie den Befehl `curl` ausführen, wie im folgenden Beispiel gezeigt:

Example

```
curl -v  http://s3.dualstack.us-west-2.amazonaws.com/
```

Sie erhalten Informationen wie im folgenden Beispiel gezeigt zurück. Wenn Sie über IPv6 verbunden sind, ist die verbundene IP-Adresse eine IPv6-Adresse.

```
* About to connect() to s3-us-west-2.amazonaws.com port 80 (#0)
*   Trying IPv6 address... connected
* Connected to s3.dualstack.us-west-2.amazonaws.com (IPv6 address) port 80 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1 OpenSSL/1.0.1t
 zlib/1.2.3
> Host: s3.dualstack.us-west-2.amazonaws.com
```

Wenn Sie Microsoft Windows 7 oder 10 verwenden, können sie testen, ob Sie über IPv6 oder IPv4 auf einen Dual-Stack-Endpunkt zugreifen können, indem Sie den Befehl `ping` wie im folgenden Beispiel gezeigt ausführen:

```
ping ipv6.s3.dualstack.us-west-2.amazonaws.com
```

Verwenden von Amazon-S3-Dual-Stack-Endpunkten

Amazon-S3-Dual-Stack-Endpunkte unterstützen Anforderungen an S3-Buckets über IPv6 und IPv4. In diesem Abschnitt wird die Verwendung von Dual-Stack-Endpunkten beschrieben.

Themen

- Amazon-S3-Dual-Stack-Endpunkte
- Verwenden von Dual-Stack-Endpunkten von der AWS CLI
- Dual-Stack-Endpunkte von der AWS-SDKs verwenden
- Verwenden von Dual-Stack-Endpunkte von der REST-API

Amazon-S3-Dual-Stack-Endpunkte

Wenn Sie eine Anforderung an einen Dual-Stack-Endpunkt richten, wird die Bucket-URL in eine IPv6- oder eine IPv4-Adresse aufgelöst. Weitere Informationen zum Zugriff auf einen Bucket über IPv6 finden Sie unte Stellen von Anforderungen an Amazon S3 über IPv6.

Wenn Sie die REST-API verwenden, können Sie direkt auf einen Amazon-S3-Endpunkt zugreifen, indem Sie den Endpunktnamen (URI) verwenden. Über einen Dual-Stack-Endpunkt können Sie auf einen S3-Bucket zugreifen, indem Sie einen Virtual-Hosted-Style- oder Path-Style-Endpunktnamen verwenden. Amazon S3 unterstützt nur regionale Dual-Stack-Endpunktnamen, d. h. Sie müssen die Region als Teil des Namens angeben.

Verwenden Sie die folgenden Namenskonventionenfür Endpunktnamen im Virtual-Hosted-Style und im Path-Style:

- Dual-Stack-Endpunkte im Virtual-Hosted–Style:

  bucketname.s3.dualstack.*aws-region*.amazonaws.com

- Dual-Stack-Endpunkt im Path–Style:

  s3.dualstack.*aws-region*.amazonaws.com/*bucketname*

Weitere Informationen zum Stil von Endpunktnamen finden Sie unter Zugreifen auf einen Amazon-S3-Bucket und Auflisten des Buckets. Eine Liste der Amazon-S3-Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.

> **Important**
>
> Für Dual-Stack-Endpunkte kann eine Transfer Acceleration verwendet werden. Weitere Informationen finden Sie unter Erste Schritte mit Amazon S3 Transfer Acceleration.

> **Note**
>
> Die beiden Arten von VPC-Endpunkten für den Zugriff auf Amazon S3 (Schnittstellen-PC-Endpunkte und Gateway-VPC-Endpunkte) bieten keine Dual-Stack-Unterstützung. Weitere Informationen zu VPC-Endpunkten für Amazon S3 finden Sie unter AWS PrivateLink für Amazon S3.

Wenn Sie die AWS Command Line Interface (AWS CLI) und AWS SDKs verwenden, können Sie einen Parameter oder ein Flag verwenden, um zu einem Dual-Stack-Endpunkt zu wechseln. Sie können den Dual-Stack-Endpunkt auch direkt als Override des Amazon-S3-Endpunkts in der Konfigurationsdatei angeben. In den folgenden Abschnitten wird erläutert, wie Dual-Stack-Endpunkte von der AWS CLI und den AWS-SDKs verwendet werden.

Verwenden von Dual-Stack-Endpunkten von der AWS CLI

Dieser Abschnitt enthält Beispiele für AWS CLI-Befehle für Anfragen an einen Dual-Stack-Endpunkt. Weitere Informationen zum Einrichten der AWS CLI finden Sie unter Entwickeln mit Amazon S3 über die AWS CLI.

Sie setzen den Konfigurationswerte `use_dualstack_endpoint` auf `true` in einem Profil in Ihrer AWS Config-Datei, um alle Amazon-S3-Anfragen von den Befehlen `s3` und `s3api` AWS CLI an die Dual-Stack-Endpunkte für die angegebene Region weiterzuleiten. Sie geben die Region in der Konfigurationsdatei oder in einem Befehl mit der Option `--region` an.

Bei Verwendung von Dual-Stack-Endpunkten mit der AWS CLI werden sowohl das Adressenformat `path` als auch das Adressenformat `virtual` unterstützt. Der Adressierungsstil, der in der Konfigurationsdatei festgelegt wird, steuert, ob der Bucketname im Hostnamen enthalten oder Teil

der URL ist. Standardmäßig versucht die CLI, den virtuellen Stil zu verwenden, wann immer das möglich ist, verwendet aber auch den Pfadstil, wenn das notwendig ist. Weitere Informationen finden Sie unter AWS CLI-Amazon-S3-Konfiguration.

Sie können auch Konfigurationsänderungen über einen Befehl vornehmen, wie im folgenden Beispiel gezeigt, das im Standardprofil `use_dualstack_endpoint` auf `true` und `addressing_style` auf `virtual` setzt.

```
$ aws configure set default.s3.use_dualstack_endpoint true
$ aws configure set default.s3.addressing_style virtual
```

Wenn Sie einen Dual-Stack-Endpunkt nur für bestimmte AWS CLI-Befehle (nicht für alle) verwenden wollen, können Sie eine der folgenden Methoden anwenden:

- Sie können den Dual-Stack-Endpunkt pro Befehl verwenden, indem Sie den Parameter `--endpoint-url` auf `https://s3.dualstack.`*`aws-region`*`.amazonaws.com` oder `http://s3.dualstack.`*`aws-region`*`.amazonaws.com` für jeden `s3`- oder `s3api`-Befehl setzen.

  ```
  $ aws s3api list-objects --bucket bucketname --endpoint-url https://s3.dualstack.aws-region.amazonaws.com
  ```

- Sie können separate Profile in Ihrer AWS Config-Datei einrichten. Legen Sie beispielsweise ein Profil an, das `use_dualstack_endpoint` auf `true` setzt, und ein Profil, das `use_dualstack_endpoint` nicht setzt. Wenn Sie einen Befehl ausführen, geben Sie an, welches Profil Sie verwenden wollen, abhängig davon, ob Sie den Dual-Stack-Endpunkt verwenden wollen oder nicht.

> **Note**
>
> Wenn Sie die AWS CLI verwenden, können Sie derzeit für Dual-Stack-Endpunkte keine Transfer Acceleration verwenden. Die AWS CLI wird jedoch demnächst unterstützt. Weitere Informationen finden Sie unter Verwenden der AWS CLI.

Dual-Stack-Endpunkte von der AWS-SDKs verwenden

Dieser Abschnitt enthält Beispiele für den Zugriff auf einen Dual-Stack-Endpunkt unter Verwendung der AWS-SDKs.

AWS SDK for JavaBeispiel für einen -Dual-Stack-Endpunkt

Das folgende Beispiel veranschaulicht, wie Sie beim Erstellen eines Amazon-S3-Clients mit dem AWS SDK for Java Dual-Stack-Endpunkte aktivieren.

Anweisungen zum Erstellen und Testen eines funktionierenden Java-Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;

public class DualStackEndpoints {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            // Create an Amazon S3 client with dual-stack endpoints enabled.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .withDualstackEnabled(true)
                    .build();

            s3Client.listObjects(bucketName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Wenn Sie das AWS SDK for Java unter Windows einsetzen, müssen Sie möglicherweise die folgende JVM (Java Virtual Machine)-Eigenschaft festlegen:

```
java.net.preferIPv6Addresses=true
```

AWSBeispiel für einen -.NET SDK Dual-Stack-Endpunkt

Wenn Sie das AWS-SDK für .NET verwenden, verwenden die Klasse `AmazonS3Config`, um die Verwendung eines Dual-Stack-Endpunkts zu erlauben, wie im folgenden Beispiel gezeigt.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DualStackEndpointTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            var config = new AmazonS3Config
            {
                UseDualstackEndpoint = true,
                RegionEndpoint = bucketRegion
            };
            client = new AmazonS3Client(config);
            Console.WriteLine("Listing objects stored in a bucket");
            ListingObjectsAsync().Wait();
        }

        private static async Task ListingObjectsAsync()
        {
            try
            {
                var request = new ListObjectsV2Request
                {
```

```
                    BucketName = bucketName,
                    MaxKeys = 10
                };
                ListObjectsV2Response response;
                do
                {
                    response = await client.ListObjectsV2Async(request);

                    // Process the response.
                    foreach (S3Object entry in response.S3Objects)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }
                    Console.WriteLine("Next Continuation Token: {0}",
 response.NextContinuationToken);
                    request.ContinuationToken = response.NextContinuationToken;
                } while (response.IsTruncated == true);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Ein vollständiges .NET-Beispiel für die Auflistung von Objekten finden Sie unter Programmgesteuertes Auflisten von Objektschlüsseln.

Weitere Informationen zum Erstellen und Testen eines funktionierenden .NET-Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

Verwenden von Dual-Stack-Endpunkte von der REST-API

Weitere Informationen über Anfragen an Dual-Stack-Endpunkte über die REST-API finden Sie unter Senden von Anforderungen an Dual-Stack-Endpunkte unter Verwendung der REST-API.

Senden von Anfragen unter Verwendung der AWS-SDKs

Themen

Sie können mit dem AWS-SDK oder durch direkte REST-API-Aufrufe innerhalb Ihrer Anwendung authentifizierte Anfragen an Amazon S3 senden. Die AWS-SDK-API verwendet die Anmeldeinformationen, die Sie für die Berechnung der Signatur für die Authentifizierung bereitstellen. Bei direkter Verwendung der REST-API in Ihren Anwendungen müssen Sie den benötigten Code zur Berechnung der Signatur für die Authentifizierung Ihrer Anforderung schreiben. Die Liste der verfügbaren AWS-SDKs finden Sie unter Sample Code and Libraries (Beispiel-Code und Bibliotheken).

Anfragen unter Verwendung von Anmeldeinformationen von AWS-Konto oder von IAM-Benutzern

Sie können die Sicherheitsanmeldeinformationen Ihres AWS-Konto oder eines IAM-Benutzers verwenden, um authentifizierte Anfragen an Amazon S3 zu senden. In diesem Abschnitt werden Beispiele aufgeführt, wie Sie über AWS SDK for Java, AWS SDK for .NET und AWS SDK for PHP authentifizierte Anfragen senden können. Die Liste der verfügbaren AWS-SDKs finden Sie unter Sample Code and Libraries (Beispiel-Code und Bibliotheken).

Jedes dieser AWS-SDKs verwendet eine SDK-spezifische Anmeldeinformationen-Anbieterkette, um die Anmeldeinformationen zu finden und zu verwenden und Aktionen für den Eigentümer der Anmeldeinformationen auszuführen. Was alle diese Anmeldeinformationen-Anbieterketten gemeinsam haben, ist, dass sie alle nach Ihrer lokalen Datei mit AWS-Anmeldeinformationen suchen.

Weitere Informationen finden Sie in den folgenden Themen:

Themen

Erstellen einer lokalen AWS-Anmeldeinformations-Datei

Die einfachste Methode zum Konfigurieren von Anmeldeinformationen für Ihre AWS-SDKs ist die Verwendung einer AWS-Anmeldeinformations-Datei. Wenn Sie die AWS Command Line Interface (AWS CLI) verwenden, wurde möglicherweise bereits eine lokale AWS-Anmeldeinformations-Datei konfiguriert. Andernfalls gehen Sie wie folgt vor, um eine Anmeldeinformationsdatei einzurichten:

1. Melden Sie sich bei der AWS Management Console an, und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Erstellen Sie einen neuen Benutzer mit Berechtigungen, die auf die Services und Aktionen beschränkt sind, auf die Ihr Code Zugriff hat. Weitere Informationen zum Erstellen eines neuen Benutzers finden Sie unter Erstellen von IAM-Benutzern (Konsole). Folgen Sie der Anleitung bis Schritt 8.
3. Wählen Sie Download .csv (CSV herunterladen) aus, um eine lokale Kopie Ihrer AWS-Anmeldeinformationen zu speichern.
4. Gehen Sie auf Ihrem Computer zu Ihrem Stammverzeichnis und erstellen Sie ein `.aws` Verzeichnis. Auf Unix-basierten Systemen (wie Linux oder OS X) befindet es sich an der folgenden Position:

   ```
   ~/.aws
   ```

   Unter Windows befindet es sich an der folgenden Position:

   ```
   %HOMEPATH%\.aws
   ```

5. Erstellen Sie in dem Verzeichnis `.aws` eine neue Datei namens `credentials`.
6. Öffnen Sie die `.csv`-Datei mit den Anmeldeinformationen aus, die Sie von der IAM-Konsole heruntergeladen haben, und kopieren Sie ihren Inhalt in die `credentials`-Datei mit dem folgenden Format:

   ```
   [default]
   aws_access_key_id = your_access_key_id
   aws_secret_access_key = your_secret_access_key
   ```

7. Speichern Sie die `credentials`-Datei, und löschen Sie die `.csv`-Datei, die Sie in Schritt 3 heruntergeladen haben.

Die gemeinsame Anmeldeinformations-Datei ist jetzt auf Ihrem lokalen Computer konfiguriert und kann für die AWS-SDKs verwendet werden.

Senden von authentifizierten Anfragen mit den AWS-SDKs

Verwenden Sie die AWS-SDKs, um authentifizierte Anfragen zu senden. Weitere Informationen zum Senden authentifizierter Anfragen finden Sie unter AWS-Sicherheitsanmeldeinformationen oder Authentifizierung von IAM Identity Center.

Java

Gehen Sie zum Senden authentifizierter Anfragen an Amazon S3 unter Verwendung der Anmeldeinformationen Ihres AWS-Konto oder eines IAM-Benutzers wie folgt vor:

- Erstellen Sie mit der `AmazonS3ClientBuilder`-Klasse eine `AmazonS3Client`-Instance.
- Führen Sie eine der `AmazonS3Client`-Methoden aus, um Anfragen an Amazon S3 zu senden. Der Client erstellt aus den von Ihnen angegebenen Anmeldeinformationen die erforderliche Signatur und nimmt sie in die Anforderung auf.

Das folgende Beispiel führt die vorhergehenden Aufgaben aus. Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsRequest;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.s3.model.S3ObjectSummary;

import java.io.IOException;
import java.util.List;

public class MakingRequests {
```

```
    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Get a list of objects in the bucket, two at a time, and
            // print the name and size of each object.
            ListObjectsRequest listRequest = new
 ListObjectsRequest().withBucketName(bucketName).withMaxKeys(2);
            ObjectListing objects = s3Client.listObjects(listRequest);
            while (true) {
                List<S3ObjectSummary> summaries = objects.getObjectSummaries();
                for (S3ObjectSummary summary : summaries) {
                    System.out.printf("Object \"%s\" retrieved with size %d\n",
 summary.getKey(), summary.getSize());
                }
                if (objects.isTruncated()) {
                    objects = s3Client.listNextBatchOfObjects(objects);
                } else {
                    break;
                }
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

So senden Sie authentifizierte Anfragen unter Verwendung der Anmeldeinformationen Ihres AWS-Konto oder eines IAM-Benutzers:

- Erstellen Sie eine Instance der `AmazonS3Client`-Klasse.
- Führen Sie eine der `AmazonS3Client`-Methoden aus, um Anfragen an Amazon S3 zu senden. Der Client erstellt aus den von Ihnen angegebenen Anmeldeinformationen die erforderliche Signatur und nimmt sie in die Anforderung auf, die er an Amazon S3 sendet.

Weitere Informationen finden Sie unter Anfragen unter Verwendung von Anmeldeinformationen von AWS-Konto oder von IAM-Benutzern.

> **Note**
>
> - Sie können den `AmazonS3Client`-Client erstellen, ohne Sicherheitsanmeldeinformationen anzugeben. Anforderung, die unter Verwendung dieses Clients gesendet werden, sind anonym und haben keine Signatur. Amazon S3 gibt einen Fehler zurück, wenn Sie anonyme Anfragen für eine Ressource stellen, die nicht öffentlich verfügbar ist.
> - Sie können ein AWS-Konto erstellen und die erforderlichen Benutzer erstellen. Sie können auch Anmeldeinformationen für diese Benutzer verwalten. Sie benötigen diese Anmeldeinformationen, um die Aufgabe im folgenden Beispiel auszuführen. Weitere Informationen finden Sie unter Configure AWS credentials (Konfigurieren von AWS-Anmeldeinformationen) im AWS SDK for .NET-Entwicklerhandbuch.
>
>   Sie können Ihre Anwendung dann auch so konfigurieren, dass sie aktiv Profile und Anmeldeinformationen abruft, und diese Anmeldeinformationen dann beim Erstellen eines AWS-Service-Clients explizit verwenden. Weitere Informationen finden Sie unter Accessing credentials and profiles in an application (Zugriff auf Anmeldeinformationen und Profile in einer Anwendung) im AWS SDK for .NET-Entwicklerhandbuch.

Das folgende C#-Beispiel veranschaulicht, wie sie die vorhergehenden Aufgaben ausführen. Weitere Informationen zur Ausführung der .NET-Beispiele in diesem Handbuch sowie Anweisungen, wie Sie Ihre Anmeldeinformationen in einer Konfigurationsdatei speichern, finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

Example

```
using Amazon;
using Amazon.S3;
```

```
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class MakeS3RequestTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            using (client = new AmazonS3Client(bucketRegion))
            {
                Console.WriteLine("Listing objects stored in a bucket");
                ListingObjectsAsync().Wait();
            }
        }

        static async Task ListingObjectsAsync()
        {
            try
            {
                ListObjectsRequest request = new ListObjectsRequest
                {
                    BucketName = bucketName,
                    MaxKeys = 2
                };
                do
                {
                    ListObjectsResponse response = await
 client.ListObjectsAsync(request);
                    // Process the response.
                    foreach (S3Object entry in response.S3Objects)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }

                    // If the response is truncated, set the marker to get the next
```

```
                    // set of keys.
                    if (response.IsTruncated)
                    {
                        request.Marker = response.NextMarker;
                    }
                    else
                    {
                        request = null;
                    }
                } while (request != null);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Funktionierende Beispiele finden Sie unter Übersicht über Amazon-S3-Objekte und Bucket-Übersicht. Sie können diese Beispiele unter Verwendung der Anmeldeinformationen Ihres AWS-Konto oder eines IAM-Benutzers testen.

Um beispielsweise alle Objektschlüssel in Ihrem Bucket aufzulisten, lesen Sie unter nac Programmgesteuertes Auflisten von Objektschlüsseln.

PHP

In diesem Abschnitt wird die Verwendung einer Klasse aus Version 3 des AWS SDK for PHP zum Senden authentifizierter Anfragen unter Verwendung Ihrer AWS-Konto- oder IAM-Benutzer-Anmeldeinformationen beschrieben. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen folgen und der AWS SDK for PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel veranschaulicht, wie der Client eine Anfrage unter Verwendung Ihrer Sicherheitsanmeldeinformationen durchführt, um alle Buckets für Ihr Konto aufzulisten.

Example

```
require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
]);

// Retrieve the list of buckets.
$result = $s3->listBuckets();

try {
    // Retrieve a paginator for listing objects.
    $objects = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    echo "Keys retrieved!" . PHP_EOL;

    // Print the list of objects to the page.
    foreach ($objects as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

> **Note**
>
> Sie können den `S3Client`-Client erstellen, ohne Sicherheitsanmeldeinformationen anzugeben. Anforderung, die unter Verwendung dieses Clients gesendet werden, sind anonym und haben keine Signatur. Amazon S3 gibt einen Fehler zurück, wenn Sie anonyme Anfragen für eine Ressource stellen, die nicht öffentlich verfügbar ist. Weitere

Informationen finden Sie unter Erstellen von anonymen Clients in der AWS SDK for PHP-Dokumentation.

Funktionierende Beispiele finden Sie unter Übersicht über Amazon-S3-Objekte. Sie können diese Beispiele unter Verwendung der Anmeldeinformationen Ihres AWS-Konto oder eines IAM-Benutzers testen.

Ein Beispiel für die Auflistung der Objektschlüssel in einem Bucket finden Sie unter Programmgesteuertes Auflisten von Objektschlüsseln.

Ruby

Bevor Sie Version 3 des AWS SDK for Ruby verwenden können, um Amazon S3 aufzurufen, müssen Sie die AWS-Zugriffs-Anmeldeinformationen einrichten, die das SDK verwendet, um Ihren Zugriff auf Ihre Buckets und Objekte zu überprüfen. Wenn Sie freigegebene Anmeldeinformationen im AWS-Anmeldeinformations-Profil auf Ihrem lokalen System bereitgestellt haben, kann Version 3 des SDK für Ruby diese Anmeldeinformationen verwenden, ohne dass Sie sie in Ihrem Code deklariert haben müssen. Weitere Informationen zum Einrichten freigegebener Anmeldeinformationen finden Sie unter Anfragen unter Verwendung von Anmeldeinformationen von AWS-Konto oder von IAM-Benutzern.

Der folgende Ruby-Codeausschnitt verwendet die Anmeldeinformationen in einer freigegebenen AWS-Anmeldeinformations-Datei auf einem lokalen Computer zum Authentifizieren einer Anfrage, um alle Objektschlüssel-Namen in einem bestimmten Bucket anzufordern. Es führt die folgenden Aktionen aus:

1. Erstellt eine Instance der `Aws::S3::Client`-Klasse.
2. Sendet eine Anfrage an Amazon S3 durch Auflisten der Objekte in einem Bucket mit der `list_objects_v2`-Methode von `Aws::S3::Client`. Der Client erstellt aus den Anmeldeinformationen der AWS-Anmeldeinformations-Datei auf Ihrem Computer den erforderlichen Signaturwert und nimmt ihn in die Anforderung auf, die er an Amazon S3 sendet.
3. Gibt das Array der Objektschlüsselnamen an das Terminal aus.

Example

```
# Prerequisites:
#   - An existing Amazon S3 bucket.
```

```
require "aws-sdk-s3"

# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if all operations succeed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_bucket_objects?(s3_client, 'doc-example-bucket')
def list_bucket_objects?(s3_client, bucket_name)
  puts "Accessing the bucket named '#{bucket_name}'..."
  objects = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if objects.count.positive?
    puts "The object keys in this bucket are (first 50 objects):"
    objects.contents.each do |object|
      puts object.key
    end
  else
    puts "No objects found in this bucket."
  end

  return true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
  return false
end

# Example usage:
def run_me
  region = "us-west-2"
  bucket_name = "BUCKET_NAME"
  s3_client = Aws::S3::Client.new(region: region)

  exit 1 unless list_bucket_objects?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

Wenn Sie keine lokale Datei mit AWS-Anmeldeinformationen besitzen, können Sie dennoch die `Aws::S3::Client`-Ressource erstellen und Code für Amazon-S3-Buckets und -Objekte ausführen. Anforderungen, die mit Version 3 des SDK for Ruby gesendet werden, sind anonym und haben standardmäßig keine Signatur. Amazon S3 gibt einen Fehler zurück, wenn Sie anonyme Anfragen für eine Ressource stellen, die nicht öffentlich verfügbar ist.

Sie können den obigen Codeausschnitt für SDK for Ruby-Anwendungen verwenden und erweitern, wie im folgenden, robusteren Beispiel gezeigt.

```
# Prerequisites:
#  - An existing Amazon S3 bucket.

require "aws-sdk-s3"

# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if all operations succeed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_bucket_objects?(s3_client, 'doc-example-bucket')
def list_bucket_objects?(s3_client, bucket_name)
  puts "Accessing the bucket named '#{bucket_name}'..."
  objects = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if objects.count.positive?
    puts "The object keys in this bucket are (first 50 objects):"
    objects.contents.each do |object|
      puts object.key
    end
  else
    puts "No objects found in this bucket."
  end

  return true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
  return false
end
```

```
# Example usage:
def run_me
  region = "us-west-2"
  bucket_name = "BUCKET_NAME"
  s3_client = Aws::S3::Client.new(region: region)

  exit 1 unless list_bucket_objects?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

Go

Example

Das folgende Beispiel verwendet AWS-Anmeldeinformationen, die vom SDK automatisch für Go aus der Datei mit den gemeinsam genutzten Anmeldeinformationen geladen werden.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/s3"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Storage Service
// (Amazon S3) client and list up to 10 buckets in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 s3Client := s3.NewFromConfig(sdkConfig)
 count := 10
```

```
 fmt.Printf("Let's list up to %v buckets for your account.\n", count)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  fmt.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Buckets) == 0 {
  fmt.Println("You don't have any buckets!")
 } else {
  if count > len(result.Buckets) {
   count = len(result.Buckets)
  }
  for _, bucket := range result.Buckets[:count] {
   fmt.Printf("\t%v\n", *bucket.Name)
  }
 }
}
```

Zugehörige Ressourcen

- Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer
- AWS SDK for PHP für Amazon S3 Aws\S3\S3Client-Klasse
- AWS SDK for PHP-Dokumentation:

Anfragen unter Verwendung temporärer Anmeldeinformationen für IAM-Benutzer erstellen

Ein AWS-Konto- oder IAM-Benutzer kann temporäre Sicherheitsanmeldeinformationen beantragen und damit authentifizierte Anfragen an Amazon S3 senden. Dieser Abschnitt enthält Beispiele, wie Sie das AWS SDK for Java, .NET und PHP verwenden können, um temporäre Sicherheitsanmeldeinformationen zu erhalten und diese zur Authentifizierung Ihrer Anfragen an Amazon S3 zu verwenden.

Java

Ein IAM-Benutzer oder ein AWS-Konto kann temporäre Sicherheitsanmeldeinformationen (siehe Senden von Anforderungen) über das AWS SDK for Java anfordern und sie für den Zugriff auf Amazon S3 verwenden. Diese Anmeldeinformationen laufen nach der angegebenen Sitzungsdauer ab.

Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, können Sie als Dauer, nach der die temporären Sicherheitsanmeldeinformationen angefordert werden sollen, eine Zeitspanne zwischen 15 Minuten und der maximalen Sitzungsdauer für die Rolle angeben. Weitere Informationen zu temporären Anmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch. Weitere Informationen zum Senden von Anfragen finden Sie unter Senden von Anforderungen.

So rufen Sie temporäre Sicherheitsanmeldeinformationen ab und greifen auf Amazon S3 zu

1. Erstellen Sie eine Instance der `AWSSecurityTokenService`-Klasse. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.
2. Rufen Sie die temporären Sicherheitsanmeldeinformationen für die gewünschte Rolle ab, indem Sie die `assumeRole()`-Methode des Security Token Service (STS) -Clients aufrufen.
3. Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einem `BasicSessionCredentials`-Objekt. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon-S3-Client bereitzustellen.
4. Erstellen Sie mit den temporären Sicherheitsanmeldeinformationen eine Instance der `AmazonS3Client`-Klasse. Mit diesem Client senden Sie Anfragen an Amazon S3. Wenn Sie beim Senden von Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

> **Note**
>
> Wenn Sie unter Verwendung der Sicherheitsanmeldeinformationen Ihres AWS-Konto temporäre Sicherheitsanmeldeinformationen erhalten haben, sind sie nur eine Stunde lang gültig. Sie können eine Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, um eine Sitzung anzufordern.

Das folgende Beispiel listet einen Satz von Objektschlüsseln im angegebenen Bucket auf. Das Beispiel fordert temporäre Sicherheitsanmeldeinformationen für eine Sitzung an, die dann zum Senden einer authentifizierten Anfrage an Amazon S3 verwendet werden.

Wenn Sie das Beispiel mit IAM-Benutzer-Anmeldeinformationen testen möchten, müssen Sie einen IAM-Benutzer unter Ihrem AWS-Konto erstellen. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.AssumeRoleRequest;
import com.amazonaws.services.securitytoken.model.AssumeRoleResult;
import com.amazonaws.services.securitytoken.model.Credentials;

public class MakingRequestsWithIAMTempCredentials {
    public static void main(String[] args) {
        String clientRegion = "*** Client region ***";
        String roleARN = "*** ARN for role to be assumed ***";
        String roleSessionName = "*** Role session name ***";
        String bucketName = "*** Bucket name ***";
```

```
       try {
           // Creating the STS client is part of your trusted code. It has
           // the security credentials you use to obtain temporary security
credentials.
           AWSSecurityTokenService stsClient =
AWSSecurityTokenServiceClientBuilder.standard()
                   .withCredentials(new ProfileCredentialsProvider())
                   .withRegion(clientRegion)
                   .build();

           // Obtain credentials for the IAM role. Note that you cannot assume the
role of
           // an AWS root account;
           // Amazon S3 will deny access. You must use credentials for an IAM user
or an
           // IAM role.
           AssumeRoleRequest roleRequest = new AssumeRoleRequest()
                   .withRoleArn(roleARN)
                   .withRoleSessionName(roleSessionName);
           AssumeRoleResult roleResponse = stsClient.assumeRole(roleRequest);
           Credentials sessionCredentials = roleResponse.getCredentials();

           // Create a BasicSessionCredentials object that contains the credentials
you
           // just retrieved.
           BasicSessionCredentials awsCredentials = new BasicSessionCredentials(
                   sessionCredentials.getAccessKeyId(),
                   sessionCredentials.getSecretAccessKey(),
                   sessionCredentials.getSessionToken());

           // Provide temporary security credentials so that the Amazon S3 client
           // can send authenticated requests to Amazon S3. You create the client
           // using the sessionCredentials object.
           AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                   .withCredentials(new
AWSStaticCredentialsProvider(awsCredentials))
                   .withRegion(clientRegion)
                   .build();

           // Verify that assuming the role worked and the permissions are set
correctly
           // by getting a set of object keys from the bucket.
           ObjectListing objects = s3Client.listObjects(bucketName);
```

```
            System.out.println("No. of Objects: " +
 objects.getObjectSummaries().size());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Ein IAM-Benutzer oder ein AWS-Konto kann temporäre Sicherheitsanmeldeinformationen über das AWS SDK for .NET anfordern und sie für den Zugriff auf Amazon S3 verwenden. Diese Anmeldeinformationen laufen nach Ende der Sitzung ab.

Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, können Sie als Dauer, nach der die temporären Sicherheitsanmeldeinformationen angefordert werden sollen, eine Zeitspanne zwischen 15 Minuten und der maximalen Sitzungsdauer für die Rolle angeben. Weitere Informationen zu temporären Anmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch. Weitere Informationen zum Senden von Anfragen finden Sie unter Senden von Anforderungen.

So rufen Sie temporäre Sicherheitsanmeldeinformationen ab und greifen auf Amazon S3 zu

1. Erstellen Sie eine Instance des AWS Security Token Service-Clients, `AmazonSecurityTokenServiceClient`. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.
2. Starten Sie eine Sitzung durch Aufruf der Methode `GetSessionToken` des STS-Clients, den Sie im vorigen Schritt erstellt haben. Sie stellen für diese Methode Sitzungsdaten mithilfe eines `GetSessionTokenRequest`-Objekts bereit.

   Die Methode gibt Ihre temporären Sicherheitsanmeldeinformationen zurück.

3. Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einer Instance des Objekts `SessionAWSCredentials`. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon-S3-Client bereitzustellen.
4. Erstellen Sie eine Instance der Klasse `AmazonS3Client`, indem Sie die temporären Sicherheitsanmeldeinformationen übergeben. Mit diesem Client senden Sie Anfragen an Amazon S3. Wenn Sie beim Senden der Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

> **Note**
>
> Wenn Sie unter Verwendung der Sicherheitsanmeldeinformationen Ihres AWS-Konto temporäre Sicherheitsanmeldeinformationen erhalten haben, sind diese Anmeldeinformationen nur eine Stunde lang gültig. Sie können die Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, um eine Sitzung anzufordern.

Das folgende C#-Beispiel listet Objektschlüssel im angegebenen Bucket auf. Zur Veranschaulichung fordert das Beispiel temporäre Sicherheitsanmeldeinformationen für die standardmäßige, einstündige Sitzung an, die dann zum Senden einer authentifizierten Anfrage an Amazon S3 verwendet werden.

Wenn Sie das Beispiel mit IAM-Benutzer-Anmeldeinformationen testen möchten, müssen Sie einen IAM-Benutzer unter Ihrem AWS-Konto erstellen. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch. Weitere Informationen zum Senden von Anfragen finden Sie unter Senden von Anforderungen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
```

```
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TempCredExplicitSessionStartTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            ListObjectsAsync().Wait();
        }

        private static async Task ListObjectsAsync()
        {
            try
            {
                // Credentials use the default AWS SDK for .NET credential search
 chain.
                // On local development machines, this is your default profile.
                Console.WriteLine("Listing objects stored in a bucket");
                SessionAWSCredentials tempCredentials = await
 GetTemporaryCredentialsAsync();

                // Create a client by providing temporary security credentials.
                using (s3Client = new AmazonS3Client(tempCredentials, bucketRegion))
                {
                    var listObjectRequest = new ListObjectsRequest
                    {
                        BucketName = bucketName
                    };
                    // Send request to Amazon S3.
                    ListObjectsResponse response = await
 s3Client.ListObjectsAsync(listObjectRequest);
                    List<S3Object> objects = response.S3Objects;
                    Console.WriteLine("Object count = {0}", objects.Count);
                }
            }
            catch (AmazonS3Exception s3Exception)
            {
```

```
                Console.WriteLine(s3Exception.Message, s3Exception.InnerException);
            }
            catch (AmazonSecurityTokenServiceException stsException)
            {
                Console.WriteLine(stsException.Message,
stsException.InnerException);
            }
        }

        private static async Task<SessionAWSCredentials>
GetTemporaryCredentialsAsync()
        {
            using (var stsClient = new AmazonSecurityTokenServiceClient())
            {
                var getSessionTokenRequest = new GetSessionTokenRequest
                {
                    DurationSeconds = 7200 // seconds
                };

                GetSessionTokenResponse sessionTokenResponse =
                            await
stsClient.GetSessionTokenAsync(getSessionTokenRequest);

                Credentials credentials = sessionTokenResponse.Credentials;

                var sessionCredentials =
                    new SessionAWSCredentials(credentials.AccessKeyId,
                                              credentials.SecretAccessKey,
                                              credentials.SessionToken);
                return sessionCredentials;
            }
        }
    }
}
```

PHP

Dieses Beispiel setzt voraus, dass Sie die Anweisungen für Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen befolgen und das AWS SDK for PHP ordnungsgemäß installiert ist.

Ein IAM–Benutzer oder ein AWS-Konto können unter Verwendung von Version 3 des AWS SDK for PHP temporäre Sicherheitsanmeldeinformationen anfordern. Mit diesen temporären

Anmeldeinformationen kann der Benutzer/das Konto dann auf Amazon S3 zugreifen. Die Anmeldeinformationen laufen mit dem Ende der Sitzung ab.

Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, können Sie als Dauer, nach der die temporären Sicherheitsanmeldeinformationen angefordert werden sollen, eine Zeitspanne zwischen 15 Minuten und der maximalen Sitzungsdauer für die Rolle angeben. Weitere Informationen zu temporären Anmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch. Weitere Informationen zum Senden von Anfragen finden Sie unter Senden von Anforderungen.

> **Note**
>
> Wenn Sie unter Verwendung Ihrer AWS-Konto-Sicherheitsanmeldeinformationen temporäre Sicherheitsanmeldeinformationen erhalten haben, sind sie lediglich eine Stunde lang gültig. Sie können eine Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, um eine Sitzung anzufordern.

Example

Das folgende PHP-Beispiel listet die Objektschlüssel im angegebenen Bucket bei Verwendung temporärer Sicherheitsanmeldeinformationen auf. Das Beispiel fordert temporäre Sicherheitsanmeldeinformationen für eine standardmäßige, einstündige Sitzung an, die dann zum Senden einer authentifizierten Anfrage an Amazon S3 verwendet werden. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen.

Wenn Sie das Beispiel mit IAM-Benutzer-Anmeldeinformationen testen möchten, müssen Sie einen IAM-Benutzer unter Ihrem AWS-Konto erstellen. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer und Administratorengruppe im IAM-Benutzerhandbuch. Ein Beispiel für das Festlegen der Sitzungsdauer beim Anfordern einer Sitzung mit IAM-Benutzer-Anmeldeinformationen finden Sie unter Anfragen unter Verwendung temporärer Anmeldeinformationen für IAM-Benutzer erstellen .

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
```

```
$bucket = '*** Your Bucket Name ***';

$sts = new StsClient([
    'version' => 'latest',
    'region' => 'us-east-1'
]);

$sessionToken = $sts->getSessionToken();

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
    'credentials' => [
        'key'    => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token'  => $sessionToken['Credentials']['SessionToken']
    ]
]);

$result = $s3->listBuckets();

try {
    // Retrieve a paginator for listing objects.
    $objects = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    echo "Keys retrieved!" . PHP_EOL;

    // List objects
    foreach ($objects as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Ruby

Ein IAM-Benutzer oder ein AWS-Konto kann temporäre Sicherheitsanmeldeinformationen über das AWS SDK for Ruby anfordern und sie für den Zugriff auf Amazon S3 verwenden. Diese Anmeldeinformationen laufen nach Ende der Sitzung ab.

Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, können Sie als Dauer, nach der die temporären Sicherheitsanmeldeinformationen angefordert werden sollen, eine Zeitspanne zwischen 15 Minuten und der maximalen Sitzungsdauer für die Rolle angeben. Weitere Informationen zu temporären Anmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch. Weitere Informationen zum Senden von Anfragen finden Sie unter Senden von Anforderungen.

> **Note**
>
> Wenn Sie unter Verwendung Ihrer AWS-Konto-Sicherheitsanmeldeinformationen temporäre Sicherheitsanmeldeinformationen erhalten haben, sind sie lediglich eine Stunde lang gültig. Sie können die Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzer-Anmeldeinformationen verwenden, um eine Sitzung anzufordern.

Das folgende Ruby-Beispiel erstellt einen temporären Benutzer, damit dieser für eine Stunde die Elemente in einem bestimmten Bucket auflisten kann. Um dieses Beispiel verwenden zu können, müssen Sie über AWS-Anmeldeinformationen mit den notwendigen Berechtigungen zum Erstellen neuer AWS Security Token Service (AWS STS) Clients und zum Auflisten von Amazon-S3-Buckets verfügen.

```
# Prerequisites:
# - A user in AWS Identity and Access Management (IAM). This user must
#   be able to assume the following IAM role. You must run this code example
#   within the context of this user.
# - An existing role in IAM that allows all of the Amazon S3 actions for all of the
#   resources in this code example. This role must also trust the preceding IAM
 user.
# - An existing S3 bucket.

require "aws-sdk-core"
require "aws-sdk-s3"
require "aws-sdk-iam"

# Checks whether a user exists in IAM.
#
# @param iam [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
```

```
# @return [Boolean] true if the user exists; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   exit 1 unless user_exists?(iam_client, 'my-user')
def user_exists?(iam_client, user_name)
  response = iam_client.get_user(user_name: user_name)
  return true if response.user.user_name
rescue Aws::IAM::Errors::NoSuchEntity
  # User doesn't exist.
rescue StandardError => e
  puts "Error while determining whether the user " \
    "'#{user_name}' exists: #{e.message}"
end

# Creates a user in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [AWS:IAM::Types::User] The new user.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   user = create_user(iam_client, 'my-user')
#   exit 1 unless user.user_name
def create_user(iam_client, user_name)
  response = iam_client.create_user(user_name: user_name)
  return response.user
rescue StandardError => e
  puts "Error while creating the user '#{user_name}': #{e.message}"
end

# Gets a user in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [AWS:IAM::Types::User] The existing user.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   user = get_user(iam_client, 'my-user')
#   exit 1 unless user.user_name
def get_user(iam_client, user_name)
  response = iam_client.get_user(user_name: user_name)
  return response.user
rescue StandardError => e
  puts "Error while getting the user '#{user_name}': #{e.message}"
```

```
end

# Checks whether a role exists in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_name [String] The role's name.
# @return [Boolean] true if the role exists; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   exit 1 unless role_exists?(iam_client, 'my-role')
def role_exists?(iam_client, role_name)
  response = iam_client.get_role(role_name: role_name)
  return true if response.role.role_name
rescue StandardError => e
  puts "Error while determining whether the role " \
    "'#{role_name}' exists: #{e.message}"
end

# Gets credentials for a role in IAM.
#
# @param sts_client [Aws::STS::Client] An initialized AWS STS client.
# @param role_arn [String] The role's Amazon Resource Name (ARN).
# @param role_session_name [String] A name for this role's session.
# @param duration_seconds [Integer] The number of seconds this session is valid.
# @return [AWS::AssumeRoleCredentials] The credentials.
# @example
#   sts_client = Aws::STS::Client.new(region: 'us-west-2')
#   credentials = get_credentials(
#     sts_client,
#     'arn:aws:iam::123456789012:role/AmazonS3ReadOnly',
#     'ReadAmazonS3Bucket',
#     3600
#   )
#   exit 1 if credentials.nil?
def get_credentials(sts_client, role_arn, role_session_name, duration_seconds)
  Aws::AssumeRoleCredentials.new(
    client: sts_client,
    role_arn: role_arn,
    role_session_name: role_session_name,
    duration_seconds: duration_seconds
  )
rescue StandardError => e
  puts "Error while getting credentials: #{e.message}"
end
```

```
# Checks whether a bucket exists in Amazon S3.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The name of the bucket.
# @return [Boolean] true if the bucket exists; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless bucket_exists?(s3_client, 'doc-example-bucket')
def bucket_exists?(s3_client, bucket_name)
  response = s3_client.list_buckets
  response.buckets.each do |bucket|
    return true if bucket.name == bucket_name
  end
rescue StandardError => e
  puts "Error while checking whether the bucket '#{bucket_name}' " \
    "exists: #{e.message}"
end

# Lists the keys and ETags for the objects in an Amazon S3 bucket.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if the objects were listed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_objects_in_bucket?(s3_client, 'doc-example-bucket')
def list_objects_in_bucket?(s3_client, bucket_name)
  puts "Accessing the contents of the bucket named '#{bucket_name}'..."
  response = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if response.count.positive?
    puts "Contents of the bucket named '#{bucket_name}' (first 50 objects):"
    puts "Name => ETag"
    response.contents.each do |obj|
      puts "#{obj.key} => #{obj.etag}"
    end
  else
    puts "No objects in the bucket named '#{bucket_name}'."
  end
  return true
```

```
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
end
```

Zugehörige Ressourcen

- Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer
- AWS SDK for PHP für Amazon S3 Aws\S3\S3Client-Klasse
- AWS SDK for PHP-Dokumentation:

Anforderungen unter Verwendung temporärer Anmeldeinformationen verbundener Benutzer

Sie können temporäre Sicherheitsanmeldeinformationen anfordern und Sie für Ihre verbundenen Benutzer oder Anwendungen bereitstellen, die Zugriff auf Ihrer AWS-Ressourcen benötigen. Dieser Abschnitt enthält Beispiele dafür, wie Sie das AWS-SDK dazu verwenden können, temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer oder Anwendungen zu erlangen und mithilfe dieser Anmeldeinformationen authentifizierte Anfragen an Amazon S3 zu senden. Die Liste der verfügbaren AWS-SDKs finden Sie unter Sample Code and Libraries (Beispiel-Code und Bibliotheken).

> **Note**
>
> Sowohl das AWS-Konto als auch ein IAM-Benutzer können temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer anfordern. Jedoch sollte aus Sicherheitsgründen nur ein IAM-Benutzer mit den notwendigen Berechtigungen diese temporären Anmeldeinformationen anfordern, um sicherzustellen, dass der verbundene Benutzer höchstens die Berechtigungen des anfragenden IAM-Benutzers erhält. In einigen Anwendungen kann es nützlich sein, einen IAM-Benutzer mit den spezifischen Berechtigungen nur zu dem Zweck zu erstellen, temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen zu erteilen.

Java

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie diese temporären Anmeldeinformationen anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, mit der die Ressourcenberechtigungen beschrieben wird, die Sie erteilen möchten. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Sie können explizit einen anderen Wert für die Dauer angeben, wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern.

> **Note**
>
> Zur Gewährleistung zusätzlicher Sicherheit beim Anfordern temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen

empfehlen wir die Verwendung eines dedizierten IAM-Benutzers mit nur den dazu benötigten Zugriffsberechtigungen. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen finden Sie unter AWS Identity and Access Management: Häufig gestellte Fragen.

Gehen Sie zum Bereitstellen von Sicherheitsanmeldeinformationen und zum Senden einer authentifizierten Anfrage für den Zugriff auf Ressourcen wie folgt vor:

- Erstellen Sie eine Instance der `AWSSecurityTokenServiceClient`-Klasse. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Verwendung der AWS SDK for Java.
- Starten Sie mit dem Aufruf der Methode `getFederationToken()` des Security Token Service (STS)-Clients eine Sitzung. Stellen Sie Sitzungsinformationen wie beispielsweise den Benutzernamen und eine IAM-Richtlinie bereit, die Sie den temporären Anmeldeinformationen anfügen möchten. Optional können Sie eine Sitzungsdauer angeben. Diese Methode gibt Ihre temporären Sicherheitsanmeldeinformationen zurück.
- Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einer Instance des Objekts `BasicSessionCredentials`. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon-S3-Client bereitzustellen.
- Erstellen Sie mit den temporären Sicherheitsanmeldeinformationen eine Instance der `AmazonS3Client`-Klasse. Mit diesem Client senden Sie Anfragen an Amazon S3. Wenn Sie beim Senden der Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

Example

Das folgende Beispiel listet die Schlüssel im angegebenen S3-Bucket auf. In dem Beispiel erhalten Sie temporäre Sicherheitsanmeldeinformationen für eine zweistündige Sitzung für Ihren verbundenen Benutzer und verwenden sie, um authentifizierte Anfragen an Amazon S3 zu senden. Zum Ausführen des Beispiels müssen Sie einen IAM-Benutzer mit einer zugeordneten Richtlinie erstellen, die es dem Benutzer ermöglicht, temporäre Sicherheitsanmeldeinformationen anzufordern und Ihre AWS-Ressourcen aufzulisten. Dies wird mit der folgenden Richtlinie erreicht:

```
{
  "Statement":[{
```

```
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
}
```

Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

Nachdem ein IAM-Benutzer erstellt und die vorhergehende Richtlinie angefügt wurde, können Sie das folgende Beispiel ausführen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.S3Actions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;

import java.io.IOException;

public class MakingRequestsWithFederatedTempCredentials {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
```

```
        String bucketName = "*** Specify bucket name ***";
        String federatedUser = "*** Federated user name ***";
        String resourceARN = "arn:aws:s3:::" + bucketName;

        try {
            AWSSecurityTokenService stsClient = AWSSecurityTokenServiceClientBuilder
                    .standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            GetFederationTokenRequest getFederationTokenRequest = new
GetFederationTokenRequest();
            getFederationTokenRequest.setDurationSeconds(7200);
            getFederationTokenRequest.setName(federatedUser);

            // Define the policy and add it to the request.
            Policy policy = new Policy();
            policy.withStatements(new Statement(Effect.Allow)
                    .withActions(S3Actions.ListObjects)
                    .withResources(new Resource(resourceARN)));
            getFederationTokenRequest.setPolicy(policy.toJson());

            // Get the temporary security credentials.
            GetFederationTokenResult federationTokenResult =
stsClient.getFederationToken(getFederationTokenRequest);
            Credentials sessionCredentials = federationTokenResult.getCredentials();

            // Package the session credentials as a BasicSessionCredentials
            // object for an Amazon S3 client object to use.
            BasicSessionCredentials basicSessionCredentials = new
BasicSessionCredentials(
                    sessionCredentials.getAccessKeyId(),
                    sessionCredentials.getSecretAccessKey(),
                    sessionCredentials.getSessionToken());
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new
AWSStaticCredentialsProvider(basicSessionCredentials))
                    .withRegion(clientRegion)
                    .build();

            // To verify that the client works, send a listObjects request using
            // the temporary security credentials.
            ObjectListing objects = s3Client.listObjects(bucketName);
```

```
            System.out.println("No. of Objects = " +
 objects.getObjectSummaries().size());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie diese temporären Anmeldeinformationen anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, mit der die Ressourcenberechtigungen beschrieben wird, die Sie erteilen möchten. Standardmäßig beträgt die Dauer einer Sitzung eine Stunde. Sie können explizit einen anderen Wert für die Dauer angeben, wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern. Informationen zum Senden authentifizierter Anfragen finden Sie unter Senden von Anforderungen.

> **Note**
>
> Wenn temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen angefordert werden, empfehlen wir, für zusätzliche Sicherheit einen dedizierten IAM-Benutzer mit nur den dazu benötigten Zugriffsberechtigungen zu verwenden. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen finden Sie unter AWS Identity and Access Management: Häufig gestellte Fragen.

Dazu führen Sie die folgenden Schritte aus:

- Sie erstellen eine Instance des AWS Security Token Service-Clients, Klasse `AmazonSecurityTokenServiceClient`. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Verwendung der AWS SDK for .NET.
- Starten Sie eine Sitzung durch Aufruf der Methode `GetFederationToken` des STS-Clients. Sie müssen Sitzungsinformationen bereitstellen, wie beispielsweise den Benutzernamen und eine IAM-Richtlinie, die Sie den temporären Anmeldeinformationen zuordnen wollen. Optional können Sie eine Sitzungsdauer angeben. Diese Methode gibt Ihre temporären Sicherheitsanmeldeinformationen zurück.
- Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einer Instance des Objekts `SessionAWSCredentials`. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon-S3-Client bereitzustellen.
- Erstellen Sie eine Instance der `AmazonS3Client`-Klasse, indem Sie die temporären Sicherheitsanmeldeinformationen übergeben. Sie verwenden diesen Client zum Senden von Anfragen an Amazon S3. Wenn Sie beim Senden der Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

Example

Das folgende C#-Beispiel listet die Schlüssel im angegebenen Bucket auf. In dem Beispiel erhalten Sie temporäre Sicherheitsanmeldeinformationen für eine zweistündige Sitzung für Ihren verbundenen Benutzer (User1) und verwenden sie, um authentifizierte Anfragen an Amazon S3 zu senden.

- In dieser Übung erstellen Sie einen IAM-Benutzer mit minimalen Berechtigungen. Unter Verwendung der Anmeldeinformationen dieses IAM-Benutzers fordern Sie temporäre Anmeldeinformationen für andere an. Dieses Beispiel listet nur die Objekte in einem bestimmgen Bucket auf. Erstellen Sie einen IAM-Benutzer, dem die folgenden Richtlinie zugeordnet ist:

```
{
  "Statement":[{
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
```

```
}
```

Die Richtlinie gestattet dem IAM-Benutzer, temporäre Sicherheitsanmeldeinformationen anzufordern, und erteilt ihm nur Zugriffsberechtigungen, um Ihre AWS-Ressourcen aufzulisten. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

- Verwenden Sie die Sicherheitsanmeldeinformationen des IAM-Benutzers, um das folgende Beispiel zu testen. Das Beispiel sendet unter Verwendung temporärer Sicherheitsanmeldeinformationen eine authentifizierte Anfrage an Amazon S3. Das Beispiel gibt die folgende Richtlinie an, wenn temporäre Sicherheitsanmeldeinformationen für den verbundenen Benutzer (User1) angefordert werden, womit der Zugriff auf eine Auflistung der Objekte in einem spezifischen Bucket eingeschränkt wird (`YourBucketName`). Sie müssen die Richtlinie aktualisieren, indem Sie Ihren eigenen bestehenden Bucket-Namen angeben.

```
{
  "Statement":[
    {
      "Sid":"1",
      "Action":["s3:ListBucket"],
      "Effect":"Allow",
      "Resource":"arn:aws:s3:::YourBucketName"
    }
  ]
}
```

- Example

  Aktualisieren Sie das folgende Beispiel und geben Sie den Bucket-Namen an, den Sie in der vorherigen Zugriffsrichtlinie für verbundene Benutzer angegeben haben. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
using System.Collections.Generic;
```

```
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TempFederatedCredentialsTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            ListObjectsAsync().Wait();
        }

        private static async Task ListObjectsAsync()
        {
            try
            {
                Console.WriteLine("Listing objects stored in a bucket");
                // Credentials use the default AWS SDK for .NET credential search
 chain.
                // On local development machines, this is your default profile.
                SessionAWSCredentials tempCredentials =
                    await GetTemporaryFederatedCredentialsAsync();

                // Create a client by providing temporary security credentials.
                using (client = new AmazonS3Client(bucketRegion))
                {
                    ListObjectsRequest listObjectRequest = new
 ListObjectsRequest();
                    listObjectRequest.BucketName = bucketName;

                    ListObjectsResponse response = await
 client.ListObjectsAsync(listObjectRequest);
                    List<S3Object> objects = response.S3Objects;
                    Console.WriteLine("Object count = {0}", objects.Count);

                    Console.WriteLine("Press any key to continue...");
                    Console.ReadKey();
                }
            }
```

```
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}'
when writing an object", e.Message);
            }
        }

        private static async Task<SessionAWSCredentials>
GetTemporaryFederatedCredentialsAsync()
        {
            AmazonSecurityTokenServiceConfig config = new
AmazonSecurityTokenServiceConfig();
            AmazonSecurityTokenServiceClient stsClient =
                new AmazonSecurityTokenServiceClient(
                                            config);

            GetFederationTokenRequest federationTokenRequest =
                                    new GetFederationTokenRequest();
            federationTokenRequest.DurationSeconds = 7200;
            federationTokenRequest.Name = "User1";
            federationTokenRequest.Policy = @"{
               ""Statement"":
               [
                 {
                   ""Sid"":""Stmt1311212314284"",
                   ""Action"":[""s3:ListBucket""],
                   ""Effect"":""Allow"",
                   ""Resource"":""arn:aws:s3:::" + bucketName + @"""
                  }
               ]
             }
            ";

            GetFederationTokenResponse federationTokenResponse =
                        await
stsClient.GetFederationTokenAsync(federationTokenRequest);
            Credentials credentials = federationTokenResponse.Credentials;

            SessionAWSCredentials sessionCredentials =
```

```
                new SessionAWSCredentials(credentials.AccessKeyId,
                                          credentials.SecretAccessKey,
                                          credentials.SessionToken);
            return sessionCredentials;
        }
    }
}
```

PHP

Dieses Thema beschreibt, wie Sie Klassen aus Version 3 des AWS SDK for PHP verwenden, um temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anzufordern und damit auf in Amazon S3 gespeicherte Ressourcen zuzugreifen. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen folgen und der AWS SDK for PHP ordnungsgemäß installiert ist.

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie diese temporären Anmeldeinformationen anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, mit der die Ressourcenberechtigungen beschrieben wird, die Sie erteilen möchten. Diese Anmeldeinformationen laufen mit dem Ende der Sitzung ab. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Sie können explizit einen anderen Wert für die Dauer angeben, wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern. Weitere Informationen zu temporären Anmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch. Informationen zum Bereitstellen temporärer Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen finden Sie unter Senden von Anforderungen.

Für zusätzliche Sicherheit beim Anfordern temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen empfehlen wir die Verwendung eines dedizierten IAM-Benutzers mit nur den dazu benötigten Zugriffsberechtigungen. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen über Identitätsverbunde finden Sie unter den Häufig gestellten Fragen zu AWS Identity and Access Management.

Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen.

Example

Das folgende PHP-Beispiel listet Schlüssel im angegebenen Bucket auf. In dem Beispiel erhalten Sie temporäre Sicherheitsanmeldeinformationen für eine einstündige Sitzung für Ihren verbundenen Benutzer (User1). Sie verwenden die temporären Sicherheitsanmeldeinformationen dann zum Senden authentifizierter Anfragen an Amazon S3.

Verwenden Sie zur Gewährleistung zusätzlicher Sicherheit beim Anfordern temporärer Anmeldeinformationen für andere die Sicherheitsanmeldeinformationen eines IAM-Benutzers, der über die Berechtigung zum Anfordern temporärer Sicherheitsanmeldeinformationen verfügt. Um sicherzustellen, dass der IAM-Benutzer dem verbundenen Benutzer nur die anwendungsspezifischen Mindestberechtigungen gewährt, können Sie auch die Zugriffsberechtigungen dieses IAM-Benutzers einschränken. Dieses Beispiel listet nur Objekte in einem bestimmten Bucket auf. Erstellen Sie einen IAM-Benutzer, dem die folgenden Richtlinie zugeordnet ist:

```
{
  "Statement":[{
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
}
```

Die Richtlinie gestattet dem IAM-Benutzer, temporäre Sicherheitsanmeldeinformationen anzufordern, und erteilt ihm nur Zugriffsberechtigungen, um Ihre AWS-Ressourcen aufzulisten. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

Jetzt können Sie die Sicherheitsanmeldeinformationen des IAM-Benutzers verwenden, um das folgende Beispiel zu testen. Das Beispiel sendet unter Verwendung temporärer Sicherheitsanmeldeinformationen eine authentifizierte Anfrage an Amazon S3. Wenn temporäre Sicherheitsanmeldeinformationen für den verbundenen Benutzer (User1) angefordert werden, gibt das Beispiel die folgenden Richtlinie an, womit der Zugriff auf eine Auflistung der Objekte in einem bestimmten Bucket eingeschränkt wird. Aktualisieren Sie die Richtlinie mit Ihrem Bucket-Namen.

```
{
```

```
  "Statement":[
    {
      "Sid":"1",
      "Action":["s3:ListBucket"],
      "Effect":"Allow",
      "Resource":"arn:aws:s3:::YourBucketName"
    }
  ]
}
```

Ersetzen Sie im folgenden Beispiel bei der Angabe der Richtlinien-Ressource `YourBucketName` durch den Namen Ihres Buckets:

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;

$bucket = '*** Your Bucket Name ***';

// In real applications, the following code is part of your trusted code. It has
// the security credentials that you use to obtain temporary security credentials.
$sts = new StsClient([
    'version' => 'latest',
    'region' => 'us-east-1'
]);

// Fetch the federated credentials.
$sessionToken = $sts->getFederationToken([
    'Name'              => 'User1',
    'DurationSeconds'    => '3600',
    'Policy'            => json_encode([
        'Statement' => [
            'Sid'               => 'randomstatementid' . time(),
            'Action'            => ['s3:ListBucket'],
            'Effect'            => 'Allow',
            'Resource'          => 'arn:aws:s3:::' . $bucket
        ]
    ])
]);

// The following will be part of your less trusted code. You provide temporary
```

```
// security credentials so the code can send authenticated requests to Amazon S3.

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
    'credentials' => [
        'key' => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token' => $sessionToken['Credentials']['SessionToken']
    ]
]);

try {
    $result = $s3->listObjects([
        'Bucket' => $bucket
    ]);
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Ruby

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie diese temporären Anmeldeinformationen vom IAM-Service anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, mit der die Ressourcenberechtigungen beschrieben werden, die Sie erteilen möchten. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie temporäre Sicherheitsanmeldeinformationen unter Verwendung von IAM-Benutzer-Anmeldeinformationen anfordern, können Sie explizit einen anderen Wert für die Gültigkeitsdauer festlegen, wenn Sie die temporären Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern. Informationen zu temporären Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen finden Sie unter Senden von Anforderungen.

> **Note**
>
> Verwenden Sie zur Gewährleistung zusätzlicher Sicherheit beim Anfordern temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen einen dedizierten IAM-Benutzer mit nur den dazu benötigten Zugriffsberechtigungen. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der

IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen finden Sie unter AWS Identity and Access Management: Häufig gestellte Fragen.

Example

Das folgende Ruby-Codebeispiel gestattet es einem verbundenen Benutzer mit einem limitierten Satz von Berechtigungen Schlüssel im angegebenen Bucket aufzulisten.

```
# Prerequisites:
#  - An existing Amazon S3 bucket.

require "aws-sdk-s3"
require "aws-sdk-iam"
require "json"

# Checks to see whether a user exists in IAM; otherwise,
# creates the user.
#
# @param iam [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [Aws::IAM::Types::User] The existing or new user.
# @example
#   iam = Aws::IAM::Client.new(region: 'us-west-2')
#   user = get_user(iam, 'my-user')
#   exit 1 unless user.user_name
#   puts "User's name: #{user.user_name}"
def get_user(iam, user_name)
  puts "Checking for a user with the name '#{user_name}'..."
  response = iam.get_user(user_name: user_name)
  puts "A user with the name '#{user_name}' already exists."
  return response.user
# If the user doesn't exist, create them.
rescue Aws::IAM::Errors::NoSuchEntity
  puts "A user with the name '#{user_name}' doesn't exist. Creating this user..."
  response = iam.create_user(user_name: user_name)
  iam.wait_until(:user_exists, user_name: user_name)
  puts "Created user with the name '#{user_name}'."
  return response.user
rescue StandardError => e
  puts "Error while accessing or creating the user named '#{user_name}':
 #{e.message}"
```

```
end

# Gets temporary AWS credentials for an IAM user with the specified permissions.
#
# @param sts [Aws::STS::Client] An initialized AWS STS client.
# @param duration_seconds [Integer] The number of seconds for valid credentials.
# @param user_name [String] The user's name.
# @param policy [Hash] The access policy.
# @return [Aws::STS::Types::Credentials] AWS credentials for API authentication.
# @example
#   sts = Aws::STS::Client.new(region: 'us-west-2')
#   credentials = get_temporary_credentials(sts, duration_seconds, user_name,
#     {
#       'Version' => '2012-10-17',
#       'Statement' => [
#         'Sid' => 'Stmt1',
#         'Effect' => 'Allow',
#         'Action' => 's3:ListBucket',
#         'Resource' => 'arn:aws:s3:::doc-example-bucket'
#       ]
#     }
#   )
#   exit 1 unless credentials.access_key_id
#   puts "Access key ID: #{credentials.access_key_id}"
def get_temporary_credentials(sts, duration_seconds, user_name, policy)
  response = sts.get_federation_token(
    duration_seconds: duration_seconds,
    name: user_name,
    policy: policy.to_json
  )
  return response.credentials
rescue StandardError => e
  puts "Error while getting federation token: #{e.message}"
end

# Lists the keys and ETags for the objects in an Amazon S3 bucket.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if the objects were listed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_objects_in_bucket?(s3_client, 'doc-example-bucket')
def list_objects_in_bucket?(s3_client, bucket_name)
```

```
  puts "Accessing the contents of the bucket named '#{bucket_name}'..."
  response = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if response.count.positive?
    puts "Contents of the bucket named '#{bucket_name}' (first 50 objects):"
    puts "Name => ETag"
    response.contents.each do |obj|
      puts "#{obj.key} => #{obj.etag}"
    end
  else
    puts "No objects in the bucket named '#{bucket_name}'."
  end
  return true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
end

# Example usage:
def run_me
  region = "us-west-2"
  user_name = "my-user"
  bucket_name = "doc-example-bucket"

  iam = Aws::IAM::Client.new(region: region)
  user = get_user(iam, user_name)

  exit 1 unless user.user_name

  puts "User's name: #{user.user_name}"
  sts = Aws::STS::Client.new(region: region)
  credentials = get_temporary_credentials(sts, 3600, user_name,
    {
      "Version" => "2012-10-17",
      "Statement" => [
        "Sid" => "Stmt1",
        "Effect" => "Allow",
        "Action" => "s3:ListBucket",
        "Resource" => "arn:aws:s3:::#{bucket_name}"
      ]
    }
  )
```

```
  exit 1 unless credentials.access_key_id

  puts "Access key ID: #{credentials.access_key_id}"
  s3_client = Aws::S3::Client.new(region: region, credentials: credentials)

  exit 1 unless list_objects_in_bucket?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

Zugehörige Ressourcen

- Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer
- AWS SDK for PHP für Amazon S3 Aws\S3\S3Client-Klasse
- AWS SDK for PHP-Dokumentation:

Senden von Anforderungen unter Verwendung der REST-API

In diesem Abschnitt finden Sie Informationen zum Senden von Anfragen an Amazon-S3-Endpunkte über die REST-API. Eine Liste der Amazon-S3-Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.

Erstellen von S3-Hostnamen für REST-API-Anforderungen

Amazon-S3-Endpunkte folgen der folgenden Struktur:

```
s3.Region.amazonaws.com
```

Amazon-S3-Zugriffspunkte-Endpunkte und Dual-Stack-Endpunkte folgen ebenfalls der Standardstruktur:

- Amazon-S3-Zugriffspunkte - `s3-accesspoint.Region.amazonaws.com`
- Dual-stack - `s3.dualstack.Region.amazonaws.com`

Eine vollständige Liste der Amazon-S3-Regionen und -Endpunkte finden Sie unter Endpunkte und Kontingente von Amazon S3 in der Allgemeine Amazon Web Services-Referenz.

Virtuell gehostete und Pfad-Stil-Anforderungen

Wenn Sie Anforderungen über die REST-API senden, können Sie für die Amazon-S3-Endpunkte URIs im virtuell gehosteten oder im Pfad-Stil verwenden. Weitere Informationen finden Sie unter Virtuelles Hosting bei Buckets.

Example Anforderung im virtuell gehosteten Stil

Nachfolgend finden Sie ein Beispiel für eine virtuell gehostete Anforderung zum Löschen der Datei `puppy.jpg` aus dem Bucket `examplebucket` in der Region USA West (Oregon). Weitere Informationen zum Wiederholen von Anforderungen finden Sie unter Anforderungen im virtuellen Hosting-Format.

```
DELETE /puppy.jpg HTTP/1.1
Host: examplebucket.s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example Pfadanforderung

Im Folgenden finden Sie ein Beispiel für eine Path-Style-Version derselben Anfrage.

```
DELETE /examplebucket/puppy.jpg HTTP/1.1
Host: s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Derzeit unterstützt Amazon S3 den URL-Zugriff in allen AWS-Regionen sowohl im virtuellem Hosting- als auch im Pfadformat. URLs im Pfadformat werden jedoch in naher Zukunft eingestellt. Weitere Informationen finden Sie im folgenden wichtigen Hinweis.

Weitere Informationen zu Anforderungen im Pfadformat finden Sie unter Anforderungen im Pfadformat.

> **Important**
>
> Update (23. September 2020) – Wir haben das Veralten von URLs im Pfadformat verschoben, um sicherzustellen, dass Kunden genügend Zeit haben, um zu URLs im

virtuellen Hosting-Format zu wechseln. Weitere Informationen finden Sie unter Amazon S3 Path Deprecation Plan – The Rest of the Story im AWS News Blog.

Senden von Anforderungen an Dual-Stack-Endpunkte unter Verwendung der REST-API

Wenn Sie die REST-API verwenden, können Sie direkt auf einen Dual-Stack-Endpunkt zugreifen, indem Sie einen Virtual-Hosted-Style- oder Path-Style-Endpunktnamen (URI) verwenden. Alle Amazon-S3-Dual-Stack-Endpunktnamen enthalten die Region. Anders als im Fall von Standardendpunkten, die nur IPv4 verwenden, verwenden sowohl Virtual-Hosted-Style- als auch Path-Style-Endpunkte regionsspezifische Endpunktnamen.

Example Dual-Stack-Endpunkt-Anforderung im virtuell gehosteten Stil

Sie können in Ihrer REST-Anforderung einen virtuell gehosteten Endpunkt wie im folgenden Beispiel gezeigt verwenden, der das Objekt `puppy.jpg` aus dem Bucket `examplebucket` in der Region USA West (Oregon) abruft.

```
GET /puppy.jpg HTTP/1.1
Host: examplebucket.s3.dualstack.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example Pfadanforderungen für Dual-Stack-Endpunkte

Sie können auch wie im folgenden Beispiel gezeigt einen Path-Style-Endpunkt in Ihrer Anfrage verwenden.

```
GET /examplebucket/puppy.jpg HTTP/1.1
Host: s3.dualstack.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Weitere Informationen zu Dual-Stack-Endpunkten finden Sie unter Verwenden von Amazon-S3-Dual-Stack-Endpunkten.

Weitere Informationen über das Senden von Anfragen mit der REST-API finden Sie in den folgenden Themen.

Themen

Virtuelles Hosting bei Buckets

Das virtuelle Hosting ist das beste Verfahren, um mehrere Websites mit einem Webserver zu unterstützen. Eine Möglichkeit, die Websites in Ihren REST-API-Anforderungen von Amazon S3 zu unterscheiden, ist die Verwendung des Hostnamens in der Anforderungs-URI anstelle des Pfadnamens in der URI. Eine gewöhnliche Amazon-S3-REST-Anfrage gibt einen Bucket an, indem sie die erste durch Schrägstrich abgetrennte Komponente des Pfads der Anfrage-URI verwendet. Stattdessen können Sie das virtuelle Hosting von Amazon S3 verwenden, um mit dem HTTP-Header `Host` einen Bucket in einem REST-API-Aufruf anzusprechen. In der Praxis interpretiert Amazon S3 `Host` so, dass die meisten Buckets automatisch (für begrenzte Anfragetypen) unter `https://`*`bucket-name`*`.s3.`*`region-code`*`.amazonaws.com` zur Verfügung stehen. Eine vollständige Liste der Amazon-S3-Regionen und -Endpunkte finden Sie unter Amazon-S3-Endpunkte und -Kontingente in der Allgemeine Amazon Web Services-Referenz.

Virtuelles Hosting hat auch andere Vorteile. Wenn Sie Ihrem Bucket denselben Namen wie Ihrer registrierten Domäne geben und diesen Namen dann zu einem DNS-Alias für Amazon S3 machen, können Sie die URL Ihrer Amazon-S3-Ressourcen vollständig anpassen, z. B, `http://my.`*`bucket-name`*`.com/`. Sie können auch im „Stammverzeichnis“ des virtuellen Servers Ihres Buckets veröffentlichen. Diese Fähigkeit kann wichtig sein, da viele vorhandene Anwendungen nach Dateien an diesem Standardspeicherort suchen. Beispielsweise wird erwartet, dass `favicon.ico`, `robots.txt` und `crossdomain.xml` im Stamm gefunden werden können.

> **Important**
>
> Bei Verwendung von virtuell gehosteten Buckets mit SSL stimmt das SSL-Platzhalterzertifikat nur mit Buckets überein, die keine Punkte (`.`) enthalten. Zur Umgehung dieser Einschränkung verwenden Sie HTTP oder schreiben Sie Ihre eigene Logik zur Verifizierung von Zertifikaten. Weitere Informationen finden Sie unter Amazon S3 Path Deprecation Plan im AWS News Blog.

Themen

Anforderungen im Pfadformat

Derzeit unterstützt Amazon S3 den URL-Zugriff in allen AWS-Regionen sowohl im virtuellem Hosting- als auch im Pfadformat. URLs im Pfadformat werden jedoch in naher Zukunft eingestellt. Weitere Informationen finden Sie im folgenden wichtigen Hinweis.

In Amazon S3 verwenden URLs, die auf Pfaden basieren, das folgende Format:

```
https://s3.region-code.amazonaws.com/bucket-name/key-name
```

Wenn Sie beispielsweise einen Bucket namens `DOC-EXAMPLE-BUCKET1` in der Region USA West (Oregon) erstellen und in dem Bucket auf das Objekt `puppy.jpg` zugreifen möchten, können Sie die folgende URL im Pfad-Stil verwenden:

```
https://s3.us-west-2.amazonaws.com/DOC-EXAMPLE-BUCKET1/puppy.jpg
```

> **Important**
>
> Update (23. September 2020) – Wir haben das Veralten von URLs im Pfadformat verschoben, um sicherzustellen, dass Kunden genügend Zeit haben, um zu URLs im virtuellen Hosting-Format zu wechseln. Weitere Informationen finden Sie unter Amazon S3 Path Deprecation Plan – The Rest of the Story im AWS News Blog.

> **Warning**
>
> Vermeiden Sie beim Hosten von Website-Inhalten, auf die über einen Web-Browser zugegriffen wird, die Verwendung von URLs im Pfadformat, da dies das Browser-Sicherheitsmodell desselben Ursprungs beeinträchtigen könnte. Für das Hosten von Website-Inhalten empfehlen wir, entweder S3-Website-Endpunkte oder eine CloudFront-Verteilung zu verwenden. Weitere Informationen finden Sie unter Website-Endpunkte und Bereitstellen einer React-basierten Single-Page-Anwendung für Amazon S3 und CloudFront in den AWS-Perspective-Guidance-Mustern.

Anforderungen im virtuellen Hosting-Format

In einer URL im virtuellen Hosting-Stil ist der Bucket-Name Teil des Domänennamens in der URL.

Amazon-S3-URLs, die auf virtuellem Hosting basieren, verwenden das folgende Format:

```
https://bucket-name.s3.region-code.amazonaws.com/key-name
```

In diesem Beispiel ist `DOC-EXAMPLE-BUCKET1` der Bucket-Name, "US West (Oregon) (USA West (Oregon))" die Region und `puppy.png` der Schlüsselname:

```
https://DOC-EXAMPLE-BUCKET1.s3.us-west-2.amazonaws.com/puppy.png
```

Bucket-Spezifikation für HTTP-**`Host`**-Header

So lange Ihre `GET`-Anforderung nicht den SSL-Endpunkt verwendet, können Sie mit dem HTTP-`Host`-Header den Bucket für die Anforderung festlegen. Der `Host`-Header in einer REST-Anforderung wird folgendermaßen interpretiert:

- Wenn der Header `Host` ausgelassen wird oder der Wert `s3.region-code.amazonaws.com` lautet, ist der Bucket für die Anforderung die erste durch Schrägstrich abgetrennte Komponente des Anforderungs-URI und der Schlüssel für die Anforderung bildet den Rest des Anforderungs-URI. Dies ist die übliche Methode wie im ersten und zweiten Beispiel in diesem Abschnitt dargestellt. Das Weglassen des `Host`-Headers ist nur bei HTTP-1.0-Anforderungen möglich.
- Wenn andernfalls der Wert des `Host`-Headers mit `.s3.region-code.amazonaws.com` endet, ist der Bucket-Name die führende Komponente im Wert des `Host`-Headers bis zu `.s3.region-code.amazonaws.com`. Der Schlüssel für die Anforderung ist die Anforderungs-URI. Diese

Interpretation stellt Buckets als Unterdomänen von `.s3.region-code.amazonaws.com` bereit (vgl. das dritte und vierte Beispiel in diesem Abschnitt).

- Ansonsten ist der Bucket für die Anforderung der klein geschriebene Wert des `Host`-Headers und die Schlüssel für die Anforderung ist die Anforderungs-URI. Diese Interpretation ist nützlich, wenn Sie als DNS-Namen Ihren Bucket-Namen registriert und diesen als kanonischen Namen (CNAME-Alias) für Amazon S3 konfiguriert haben. Das Verfahren zum Registrieren von Domänennamen und Konfigurieren von CNAME-DNS-Einträgen ist nicht Gegenstand dieses Leitfadens. Das Ergebnis wird jedoch im letzten Beispiel in diesem Abschnitt dargestellt.

Beispiele

In diesem Abschnitt finden Sie Beispiel-URLs und -anforderungen.

Example – URLs und Anforderungen im Pfadformat

In diesem Beispiel wird Folgendes verwendet:

- Bucket-Name – `example.com`
- Region: USA Ost (Nord-Virginia)
- Schlüsselname – `homepage.html`

Die URL lautet wie folgt:

```
http://s3.us-east-1.amazonaws.com/example.com/homepage.html
```

die Anforderung lautet wie folgt:

```
GET /example.com/homepage.html HTTP/1.1
Host: s3.us-east-1.amazonaws.com
```

die Anforderung mit HTTP 1.0 unter Auslassen des `Host`-Headers lautet wie folgt:

```
GET /example.com/homepage.html HTTP/1.0
```

Informationen zu mit DNS kompatiblen Namen finden Sie unter Einschränkungen. Weitere Informationen zu Schlüsseln finden Sie unter Schlüssel.

Example – URLs und Anforderungen im virtuellen Hosting-Format

In diesem Beispiel wird Folgendes verwendet:

- Bucket-Name – `DOC-EXAMPLE-BUCKET1`
- Region - Europa (Irland)
- Schlüsselname – `homepage.html`

Die URL lautet wie folgt:

```
http://DOC-EXAMPLE-BUCKET1.s3.eu-west-1.amazonaws.com/homepage.html
```

die Anforderung lautet wie folgt:

```
GET /homepage.html HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.eu-west-1.amazonaws.com
```

Example – CNAME-Aliasmethode

Um diese Methode zu verwenden, müssen Sie den DNS-Namen als CNAME-Alias für konfiguriere *`bucket-name`*`.s3.us-east-1.amazonaws.com`. Weitere Informationen finden Sie unter Anpassen von Amazon-S3-URLs mit CNAME-Einträgen.

In diesem Beispiel wird Folgendes verwendet:

- Bucket-Name – `example.com`
- Schlüsselname – `homepage.html`

Die URL lautet wie folgt:

```
http://www.example.com/homepage.html
```

Das Beispiel ist wie folgt:

```
GET /homepage.html HTTP/1.1
Host: www.example.com
```

Anpassen von Amazon-S3-URLs mit CNAME-Einträgen

Abhängig von den Anforderungen können Sie die Anzeige von `s3.region-code.amazonaws.com` auf der Website oder im Service gegebenenfalls unterbinden. Wenn Sie beispielsweise Websiteabbilder auf Amazon S3 hosten, bevorzugen Sie möglicherweise `http://images.example.com/` anstelle von `http://images.example.com.s3.us-east-1.amazonaws.com/`. Auf Buckets mit einem mit DNS kompatiblen Namen kann wie folgt verwiesen werden: `http://BucketName.s3.Region.amazonaws.com/[Filename]`, z. B. `http://images.example.com.s3.us-east-1.amazonaws.com/mydog.jpg`. Durch die Verwendung von CNAME können Sie `images.example.com` einem Amazon-S3-Host-Namen zuordnen, sodass die vorherige URL als `http://images.example.com/mydog.jpg` angezeigt wird.

Der Bucket-Name muss dem CNAME entsprechen. Wenn Sie z. B. einen CNAME für die Zuordnung von `images.example.com` zu `images.example.com.s3.us-east-1.amazonaws.com` erstellen, sind `http://images.example.com/filename` und `http://images.example.com.s3.us-east-1.amazonaws.com/filename` identisch.

Der CNAME DNS-Datensatz sollte einen Alias Ihres Domänennamens für den entsprechenden Host-Namen im Stil des virtuellen Hostings erstellen. Wenn der Bucket-Name und der Domänenname z. B. `images.example.com` lauten und sich Ihr Bucket in der Region USA Ost (Nord-Virginia) befindet, sollte der CNAME-Datensatz einen Alias für `images.example.com.s3.us-east-1.amazonaws.com` erstellen.

```
images.example.com CNAME    images.example.com.s3.us-east-1.amazonaws.com.
```

Amazon S3 verwendet den Host-Namen, um den Bucket-Namen zu ermitteln. CNAME und Bucket-Name müssen also identisch sein. Nehmen wir beispielsweise an, dass Sie `www.example.com` als CNAME für `www.example.com.s3.us-east-1.amazonaws.com` konfiguriert haben. Wenn Sie auf `http://www.example.com` zugreifen, empfängt Amazon S3 eine Anforderung, die in etwa wie folgt aussieht:

Example

```
GET / HTTP/1.1
Host: www.example.com
Date: date
Authorization: signatureValue
```

Amazon S3 sieht nur den ursprünglichen Host-Namen `www.example.com` und kennt die zum Auflösen der Anforderung verwendete CNAME-Zuordnung nicht.

In einem CNAME-Alias können Sie jeden beliebigen Amazon-S3-Endpunkt verwenden. Beispielsweise kann `s3.ap-southeast-1.amazonaws.com` in CNAME-Aliasnamen verwendet werden. Weitere Informationen zu Endpunkten finden Sie unter Anforderungsendpunkte. Informationen zum Erstellen einer statischen Website mit einer benutzerdefinierten Domäne finden Sie unter Tutorial: Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten bei Route 53 registrierten Domäne.

Important

Wenn Sie benutzerdefinierte URLs mit CNAME-Einträgen verwenden, müssen Sie sicherstellen, dass für jeden von Ihnen konfigurierten CNAME- oder Alias-Eintrag ein entsprechender Bucket vorhanden ist. Wenn Sie beispielsweise DNS-Einträge für `www.example.com` und `login.example.com` erstellen, um Webinhalte mit S3 zu veröffentlichen, müssen Sie die beiden Buckets `www.example.com` und `login.example.com` erstellen.

Wenn ein CNAME- oder Alias-Eintrag konfiguriert ist, der auf einen S3-Endpunkt ohne entsprechenden Bucket verweist, kann jeder AWS-Benutzer diesen Bucket erstellen und Inhalte unter dem konfigurierten Alias veröffentlichen, auch wenn der Eigentümer nicht derselbe ist.

Aus diesem Grund empfehlen wir, den entsprechenden CNAME oder Alias zu ändern oder zu entfernen, wenn Sie einen Bucket löschen.

Verknüpfen eines Host-Namens mit einem Amazon-S3-Bucket

So verknüpfen Sie einen Host-Namen unter Verwendung eines CNAME-Alias mit einem Amazon-S3-Bucket

1. Wählen Sie einen Host-Namen aus, der zu einer von Ihnen kontrollierten Domäne gehört.

   Dieses Beispiel verwendet die Unterdomäne `images` der Domäne `example.com`.

2. Erstellen Sie einen Bucket, der dem Host-Namen entspricht.

   In diesem Beispiel lauten der Host- und der Bucket-Name `images.example.com`. Der Bucket-Name muss exakt mit dem Host-Namen übereinstimmen.

3. Erstellen Sie einen CNAME-Eintrag, der den Host-Namen als Alias für den Amazon-S3-Bucket definiert.

   Beispiel:

   ```
   images.example.com CNAME images.example.com.s3.us-west-2.amazonaws.com
   ```

   > **Important**
   >
   > Aus Gründen des Anforderungs-Routings muss der CNAME-Eintrag genau wie im vorherigen Beispiel dargestellt definiert werden. Andernfalls kann sich trotz richtig erscheinender Funktion unvorhersehbares Verhalten ergeben.

   Das Verfahren für die Konfiguration von CNAME-DNS-Einträgen ist abhängig von Ihrem DNS-Server oder DNS-Anbieter. Spezifische Informationen finden Sie in Ihrer Serverdokumentation oder erhalten Sie von Ihrem Anbieter.

Einschränkungen

Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Abwärtskompatibilität

Die folgenden Abschnitte behandeln verschiedene Aspekte der Amazon-S3-Abwärtskompatibilität, die sich auf URL-Anforderungen im Pfad- oder virtuellen Hosting-Format beziehen.

Legacy-Endpunkte

Einige Regionen unterstützen Legacy-Endpunkte. Sie sehen diese Endpunkte möglicherweise in Ihren Serverzugriffsprotokollen oder AWS CloudTrail-Protokollen. Weitere Informationen finden Sie im folgenden Abschnitt. Eine vollständige Liste der Amazon-S3-Regionen und -Endpunkte finden Sie unter Endpunkte und Kontingente von Amazon S3 in der Allgemeine Amazon Web Services-Referenz.

Important

Obwohl Sie möglicherweise Legacy-Endpunkte in Ihren Protokollen sehen, empfehlen wir Ihnen, immer die Standardendpunktsyntax für den Zugriff auf Ihre Buckets zu verwenden. Amazon-S3-URLs, die auf virtuellem Hosting basieren, verwenden das folgende Format:

```
https://bucket-name.s3.region-code.amazonaws.com/key-name
```

In Amazon S3 verwenden URLs, die auf Pfaden basieren, das folgende Format:

```
https://s3.region-code.amazonaws.com/bucket-name/key-name
```

S3-Region

Einige ältere Amazon-S3-Regionen unterstützen Endpunkte, die einen Bindestrich (`-`) zwischen `s3` und der Region (z. B. `s3-us-west-2`) anstelle eines Punkts (z. B. `s3.us-west-2`) enthalten. Wenn sich Ihr Bucket in einer dieser Regionen befindet, wird möglicherweise das folgende Endpunktformat in Ihren Server-Zugriffsprotokollen oder CloudTrail-Protokollen angezeigt:

```
https://bucket-name.s3-region-code.amazonaws.com
```

In diesem Beispiel lautet der Bucket-Name DOC-EXAMPLE-BUCKET1 und die Region USA West (Oregon):

```
https://DOC-EXAMPLE-BUCKET1.s3-us-west-2.amazonaws.com
```

Globaler Legacy-Endpunkt

Für einige Regionen können Sie den globalen Legacy-Endpunkt verwenden, um Anforderungen zu erstellen, die keinen regionsspezifischen Endpunkt angeben. Der globale Legacy-Endpunkt lautet wie folgt:

```
bucket-name.s3.amazonaws.com
```

In Ihren Server-Zugriffsprotokollen oder CloudTrail-Logs sehen Sie möglicherweise Anforderungen, die den globalen Legacy-Endpunkt verwenden. In diesem Beispiel lautet der Bucket-Name `DOC-EXAMPLE-BUCKET1` und der globale Legacy-Endpunkt ist folgender:

```
https://DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
```

Anforderungen im virtuellen Hosting-Format für USA Ost (Nord-Virginia)

Anforderungen, die über den globalen Legacy-Endpunkt gesendet werden, werden standardmäßig an USA Ost (Nord-Virginia) weitergeleitet. Daher wird manchmal der ältere globale Endpunkt anstelle des regionalen Endpunkts für USA Ost (Nord-Virginia) verwendet. Wenn Sie einen Bucket in USA Ost (Nord-Virginia) erstellen und den globalen Endpunkt verwenden, leitet Amazon S3 Ihre Anfrage standardmäßig an diese Region weiter.

Anforderungen im virtuellem Hosting-Format für andere Regionen

Der globale Legacy-Endpunkt wird auch für Anforderungen im virtuellen Hosting-Format in anderen unterstützten Regionen verwendet. Wenn Sie einen Bucket in einer Region erstellen, die vor dem 20. März 2019 gelauncht wurde, und den globalen Legacy-Endpunkt verwenden, aktualisiert Amazon S3 den DNS-Eintrag so, dass die Anforderung an den richtigen Speicherort umgeleitet wird. Dies kann eine Weile dauern. In der Zwischenzeit wird die Standardregel angewendet und Ihre Anfrage im virtuellen Hosting-Format wird an die Region USA Ost (Nord-Virginia) weitergeleitet. Amazon S3 leitet sie dann mit einer temporären HTTP-307-Weiterleitung an die richtige Region um.

Für S3 Buckets in Regionen, die nach dem 20. März 2019 gelauncht wurden, leitet der DNS-Server die Anforderung nicht direkt an die AWS-Region weiter, in der sich der Bucket befindet. Stattdessen wird ein "HTTP 400 Bad Request"-Fehler zurückgegeben. Weitere Informationen finden Sie unter Senden von Anforderungen.

Anforderungen im Pfadformat

Für die Region USA Ost (Nord-Virginia) können Sie den globalen Legacy-Endpunkt für Anforderungen im Pfadformat verwenden.

Für alle anderen Regionen erfordert die Syntax im Pfadformat, dass beim Zugriff auf einen Bucket der regionsspezifische Endpunkt verwendet werden muss. Wenn Sie versuchen, auf einen Bucket mit dem globalen Legacy-Endpunkt oder einem anderen Endpunkt zuzugreifen, der nicht der Region entspricht, in der sich der Bucket befindet, erhalten Sie als Antwortcode einen „HTTP 307 Temporary Redirect“-Fehler und eine Meldung, die den richtigen URI für Ihre Ressource angibt. Wenn Sie beispielsweise `https://s3.amazonaws.com/`*`bucket-name`* für einen Bucket verwenden, der in der Region USA West (Oregon) erstellt wurde, wird ein „HTTP 307 Temporary Redirect“-Fehler angezeigt.

Anforderungsumleitung und die REST-API

Themen

In diesem Abschnitt wird beschrieben, wie HTTP-Umleitungen unter Verwendung der Amazon-S3-REST-API verarbeitet werden. Allgemeine Informationen zu Amazon-S3-Weiterleitungen finden Sie unter Senden von Anforderungen in der Amazon-Simple-Storage-Service-API-Referenz.

Umleitungen und HTTP-Benutzeragenten

Programme, die die Amazon S3 REST-API verwenden, sollten Umleitungen auf der Anwendungsschicht oder auf der HTTP-Ebene verarbeiten. Es können viele HTTP-Client-Bibliotheken und Benutzeragenten konfiguriert werden, um Umleitungen automatisch korrekt zu verarbeiten. Viele andere beinhalten jedoch fehlerhafte oder unvollständige Implementierungen für Umleitungen.

Bevor Sie sich darauf verlassen, dass eine Bibliothek die Umleitungsanfrage erfüllt, testen Sie die folgenden Fälle:

- Überprüfen Sie, ob alle HTTP-Anfrageheader korrekt in die Umleitungsanfrage aufgenommen wurden (die zweite Anfrage nach Empfang einer Umleitung), einschließlich der HTTP-Standards, wie beispielsweise Autorisierung und Datum.
- Überprüfen Sie, ob auch andere als GET-Umleitungen ordnungsgemäß funktionieren, wie beispielsweise PUT und DELETE.
- Überprüfen Sie, ob PUT-Anfragen den Umleitungen ordnungsgemäß folgen.
- Überprüfen Sie, ob PUT-anfragenden Umleitungen ordnungsgemäß folgen, wenn die Antwort auf 100 continue lang dauert.

HTTP-Benutzeragenten, die streng konform zu RFC 2616 arbeiten, fordern möglicherweise eine explizite Bestätigung, bevor sie einer Umleitung folgen, wenn die HTTP-Anfragemethode nicht GET oder HEAD ist. Im Allgemeinen ist es sicher, von Amazon S3 erstellten Umleitungen automatisch zu folgen, weil das System nur Umleitungen auf Hosts innerhalb der Domäne amazonaws.com ausgibt, und die Wirkung der umgeleiteten Anfrage dieselbe ist wie die der ursprünglichen Anforderung.

Umleitungen und 100 Continue

Um die Verarbeitung von Umleitungen zu vereinfachen, die Effizienzen zu verbessern und die Kosten zu vermeiden, die beim doppelten Versenden eines umgeleiteten Anfragerumpfs entstehen, konfigurieren Sie Ihre Anwendung so, dass sie 100 continues für PUT-Operationen verwendet. Wenn Ihre Anwendung 100 continue verwendet, sendet sie den Anfragerumpf erst dann, wenn sie eine Bestätigung erhält. Wird die Nachricht basierend auf den Header abgewiesen, wird der Rumpf der Meldung nicht gesendet. Weitere Informationen zu 100-continue finden Sie unter RFC 2616 Abschnitt 8.2.3.

> **Note**
>
> Laut RFC 2616 sollten Sie nicht unendlich lange warten, bevor Sie den Anfragerumpf senden, wenn Sie `Expect: Continue` für einen unbekannten HTTP-Server verwenden. Der Grund dafür ist, dass einige HTTP-Server 100 continue nicht erkennen. Amazon S3 erkennt jedoch nicht, ob Ihre Anforderung ein `Expect: Continue` enthält, und reagiert mit einem vorläufigen 100-continue-Status oder einem endgültigen Statuscode. Darüber hinaus tritt kein Umleitungscode auf, nachdem die provisorische 100-continue-Genehmigung empfangen wurde. Dies hilft Ihnen, den Empfang einer Umleitungsantwort zu vermeiden, wenn Sie noch den Anfragerumpf schreiben.

Beispiel für eine Umleitung

Dieser Abschnitt bietet ein Beispiel für eine Client/Server-Interaktion mit HTTP-Umleitungen und 100 continue.

Nachfolgend finden Sie ein Beispiel für ein PUT in den `quotes.s3.amazonaws.com`-Bucket.

```
PUT /nelson.txt HTTP/1.1
Host: quotes.s3.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000

Content-Length: 6
Expect: 100-continue
```

Amazon S3 gibt Folgendes zurück:

```
HTTP/1.1 307 Temporary Redirect
```

```
Location: http://quotes.s3-4c25d83b.amazonaws.com/nelson.txt?rk=8d47490b
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Mon, 15 Oct 2007 22:18:46 GMT

Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
  <Message>Please re-send this request to the
  specified temporary endpoint. Continue to use the
  original request endpoint for future requests.
  </Message>
  <Endpoint>quotes.s3-4c25d83b.amazonaws.com</Endpoint>
  <Bucket>quotes</Bucket>
</Error>
```

Der Client folgt der Umleitungsantwort und gibt eine neue Anfrage an den temporären Endpunkt `quotes.s3-4c25d83b.amazonaws.com` aus.

```
PUT /nelson.txt?rk=8d47490b HTTP/1.1
Host: quotes.s3-4c25d83b.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000

Content-Length: 6
Expect: 100-continue
```

Amazon S3 gibt ein 100 continue zurück, das darauf hinweist, dass der Client damit fortfahren soll, den Anforderungstext zu senden.

```
HTTP/1.1 100 Continue
```

Der Client sendet den Anfragerumpf.

```
ha ha\n
```

Amazon S3 gibt die endgültige Antwort zurück:

```
HTTP/1.1 200 OK
Date: Mon, 15 Oct 2007 22:18:48 GMT
```

```
ETag: "a2c8d6b872054293afd41061e93bc289"
Content-Length: 0
Server: AmazonS3
```

Entwickeln mit Amazon S3 über die AWS CLI

Befolgen Sie diese Schritte zum Herunterladen und Konfigurieren der AWS Command Line Interface (AWS CLI).

Eine Liste der Amazon-S3-AWS CLI-Befehle finden Sie auf den folgenden Seiten der AWS CLI-Befehlsreferenz:

- S3
- s3api
- s3control

> **Note**
>
> Services in AWS, wie z. B. Amazon S3, erfordern beim Zugriff die Eingabe von Anmeldeinformationen. Der Service kann dann feststellen, ob Sie über die Berechtigung für den Zugriff auf seine Ressourcen verfügen. Für die Konsole müssen Sie Ihr Passwort eingeben. Sie können Zugriffsschlüssel für Ihr AWS-Konto erstellen, um auf die AWS CLI oder die API zuzugreifen. Wir raten Ihnen jedoch davon ab, mittels der Anmeldeinformationen für Ihr AWS-Konto auf AWS zuzugreifen. Stattdessen empfehlen wir, AWS Identity and Access Management (IAM) zu verwenden. Erstellen Sie einen IAM-Benutzer und fügen Sie den Benutzer zu einer IAM-Gruppe mit Administrator-Berechtigungen hinzu. Anschließend gewähren Sie dem von Ihnen erstellten IAM-Benutzer administrative Berechtigungen. Sie können dann mithilfe einer speziellen URL und mit den Anmeldeinformationen für den IAM-Benutzer auf AWS zugreifen. Weitere Anweisungen finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

Einrichten von AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface-Benutzerhandbuch:

   - Einrichtung der AWS Command Line Interface

- Konfigurieren von AWS Command Line Interface

2. Fügen Sie ein benanntes Profil für den Administratorbenutzer in der AWS CLI-Konfigurationsdatei hinzu. Verwenden Sie dieses Profil beim Ausführen von AWS CLI-Befehlen. Weitere Informationen finden Sie unter Benannte Profile für die AWS CLI im AWS Command Line Interface-Benutzerhandbuch.

```
[adminuser]
aws_access_key_id = adminuser access key ID
aws_secret_access_key = adminuser secret access key
region = aws-region
```

   Eine Liste der verfügbaren AWS-Regionen finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.

3. Überprüfen Sie die Einrichtung, indem Sie die folgenden Befehle in die Befehlszeile eingeben.

   - Führen Sie den `help`-Befehl aus, um zu überprüfen, ob die AWS CLI auf Ihrem Computer installiert ist:

     ```
     aws help
     ```

   - Führen Sie einen Befehl `S3` mit den Anmeldeinformationen `adminuser` aus, die Sie soeben erstellt haben. Fügen Sie hierzu Ihrem Befehl den Parameter `--profile` hinzu, um den Profilnamen anzugeben. In diesem Beispiel listet der Befehl `ls` Buckets in Ihrem Konto auf. Die AWS CLI verwendet `adminuser`-Anmeldeinformationen, um die Anforderung zu authentifizieren.

     ```
     aws s3 ls --profile adminuser
     ```

Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer

Sie können auch die AWS-SDKs für die Entwicklung von Anwendungen mit Amazon S3 verwenden. Die AWS-SDKs vereinfachen Ihre Programmierungs-Aufgaben, indem sie die zugrunde liegende REST-API umhüllen. Die AWS Mobile SDKs und die AWS Amplify- JavaScript Bibliothek sind auch für die Erstellung verbundener mobiler und Webanwendungen mit verfügbarAWS.

In diesem Abschnitt finden Sie eine Übersicht über die Verwendung von AWS-SDKs für die Entwicklung von Amazon-S3-Anwendungen. Darüber hinaus beschreibt dieser Abschnitt, wie Sie die AWS-SDK-Codebeispiele in dieser Anleitung testen können.

Themen

Neben den AWS-SDKs stehen auch AWS-Explorer für Visual Studio und Eclipse für Java IDE zur Verfügung. In diesem Fall werden die SDKs und die Explorer zusammen als AWS-Toolkits bereitgestellt.

Sie können auch die AWS Command Line Interface (AWS CLI) für die Verwaltung von Amazon-S3-Buckets und Objekten verwenden.

AWS Toolkit for Eclipse

Die AWS Toolkit for Eclipse beinhaltet AWS SDK for Java und AWS Explorer für Eclipse. Der AWS-Explorer for Eclipse ist ein Open-Source Plug-In für die Eclipse für Java IDE, die es den Entwicklern vereinfacht, Java-Anwendungen mit AWS zu entwickeln, zu debuggen und bereitzustellen. Mit der easy-to-use GUI können Sie auf Ihre AWS Infrastruktur zugreifen und diese verwalten, einschließlich Amazon S3. Sie können allgemeine Operationen ausführen, wie beispielsweise die Verwaltung Ihrer Buckets und Objekte oder die Einrichtung von IAM-Richtlinien bei der Entwicklung von Anwendungen, und all das innerhalb des Kontexts der Eclipse for Java IDE. Weitere Informationen zur Einrichtung finden Sie unter Einrichten des Toolkits. Beispiele für die Verwendung des Explorers finden Sie unter How to Access AWS Explorer (Zugreifen auf AWS Explorer).

AWS Toolkit for Visual Studio

AWS-Explorer for Visual Studio ist eine Erweiterung von Microsoft Visual Studio, der es Entwicklern einfacher macht, .NET-Anwendungen mit Amazon Web Services zu entwickeln, zu debuggen und bereitzustellen. Mit der easy-to-use GUI können Sie auf Ihre AWS Infrastruktur zugreifen und diese verwalten, einschließlich Amazon S3. Sie können allgemeine Operationen ausführen, wie beispielsweise die Verwaltung Ihrer Buckets und Objekte oder die Einrichtung von IAM-Richtlinien bei der Entwicklung von Anwendungen, und all das innerhalb des Kontexts von Visual Studio. Weitere Informationen zur Einrichtung finden Sie unter Einrichten von AWS Toolkit for Visual Studio. Beispiele für die Verwendung von Amazon S3 unter Verwendung des Explorers finden Sie unter Using Amazon S3 from AWS Explorer (Verwenden von Amazon S3 im AWS Explorer).

AWS-SDKs

Sie können nur den SDK herunterladen. Weitere Informationen zum Herunterladen der SDK-Bibliotheken finden Sie unter Beispiel-Code-Bibliotheken.

AWS CLI

Die AWS CLI ist ein vereinheitlichtes Tool zur Verwaltung von AWS-Services, einschließlich von Amazon S3. Weitere Informationen zum Herunterladen von AWS CLI finden Sie unter AWS Command Line Interface.

Verwenden dieses Service mit einem AWS SDK

AWS-Software Development Kits (SDKs) sind für viele gängige Programmiersprachen erhältlich. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK for C++	Codebeispiele für AWS SDK for C++
AWS SDK for Go	Codebeispiele für AWS SDK for Go
AWS SDK for Java	Codebeispiele für AWS SDK for Java
AWS SDK for JavaScript	Codebeispiele für AWS SDK for JavaScript
AWS SDK for Kotlin	Codebeispiele für AWS SDK for Kotlin
AWS SDK for .NET	Codebeispiele für AWS SDK for .NET

SDK-Dokumentation	Codebeispiele
AWS SDK for PHP	Codebeispiele für AWS SDK for PHP
AWS SDK for Python (Boto3)	Codebeispiele für AWS SDK for Python (Boto3)
AWS SDK for Ruby	Codebeispiele für AWS SDK for Ruby
AWS SDK for Rust	Codebeispiele für AWS SDK for Rust
AWS SDK für SAP ABAP	Codebeispiele für AWS SDK für SAP ABAP
AWS SDK for Swift	Codebeispiele für AWS SDK for Swift

Weitere Beispiele speziell für diesen Service finden Sie unter Codebeispiele für Amazon S3 unter Verwendung von AWS SDKs.

> **Beispiel für die Verfügbarkeit**
>
> Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Feedback geben auswählen.

Angabe der Signature-Version in der Anforderungs-Authentifizierung

Amazon S3 unterstützt in den meisten AWS-Regionen nur AWS Signature Version 4. In einigen der älteren AWS-Regionen unterstützt Amazon S3 jedoch sowohl Signature Version 4 als auch Signature Version 2. Signature Version 2 wird jedoch deaktiviert (veraltet). Weitere Informationen zum Ende der Unterstützung für Signature Version 2 finden Sie unter AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet).

Eine Liste aller Amazon-S3-Regionen und der von ihnen unterstützten Signature-Versionen finden Sie unter Regions and Endpoints (Regionen und Endpunkte) in der Allgemeinen AWS-Referenz.

Für alle AWS-Regionen verwenden AWS-SDKs standardmäßig Signature Version 4, um die Anforderungen zu authentifizieren. Bei Verwendung von AWS-SDKs, die vor Mai 2016 veröffentlicht wurden, müssen Sie möglicherweise Signature Version 4 anfordern, wie in der folgenden Tabelle gezeigt.

SDK	Anforderung von Signature Version 4 für die Anforderungs-Authentifizierung
AWS CLI	Für das Standardprofil führen Sie den folgenden Befehl aus: `$ aws configure set default.s3.signature_version s3v4` Für ein benutzerdefiniertes Profil führen Sie den folgenden Befehl aus: `$ aws configure set profile.your_profile_name.s3.signature_version s3v4`
Java-SDK	Fügen Sie Ihrem Code Folgendes hinzu: `System.setProperty(SDKGlobalConfiguration.ENABLE_S3_SIGV4_SYSTEM_PROPERTY, "true");` Oder geben Sie in der Befehlszeile Folgendes an: `-Dcom.amazonaws.services.s3.enableV4`
JavaScript SDK	Setzen Sie den `signatureVersion` -Parameter auf `v4`, wenn Sie den Client konstruieren: `var s3 = new AWS.S3({signatureVersion: 'v4'});`
PHP SDK	Setzen Sie den Parameter `signature` auf `v4`, wenn Sie den Amazon S3 Service-Client für PHP SDK v2 erstellen: `<?php` `$client = S3Client::factory([` `'region' => 'YOUR-REGION',` `'version' => 'latest',` `'signature' => 'v4'` `]);`

SDK	Anforderung von Signature Version 4 für die Anforderungs-Authentifizierung
	Setzen Sie den Parameter `signature_version` bei Verwendung des PHP SDK v3 im Rahmen der Erstellung des Amazon S3 Service-Clients auf `v4`: `<?php` `$s3 = new Aws\S3\S3Client([` `'version' => '2006-03-01',` `'region' => 'YOUR-REGION',` `'signature_version' => 'v4'` `]);`
Python-Boto SDK	Geben Sie Folgendes in der boto-Standardkonfigurationsdatei an: `[s3] use-sigv4 = True`
Ruby SDK	Ruby SDK – Version 1: Setzen Sie den `:s3_signature_version` -Parameter auf `:v4`, wenn Sie den Client konstruieren: `s3 = AWS::S3::Client.new(:s3_signature_version => :v4)` Ruby SDK – Version 3: Setzen Sie den `signature_version` -Parameter auf `v4`, wenn Sie den Client konstruieren: `s3 = Aws::S3::Client.new(signature_version: 'v4')`

SDK	Anforderung von Signature Version 4 für die Anforderungs-Authentifizierung
.NET SDK	Fügen Sie den folgenden Code hinzu, bevor Sie den Amazon-S3-Client erstellen: `AWSConfigsS3.UseSignatureVersion4 = true;` Oder fügen Sie der Konfigurationsdatei Folgendes hinzu: `<appSettings>` `  <add key="AWS.S3.UseSignatureVersion4" value="true" />` `</appSettings>`

AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet)

Signature Version 2 wird unter Amazon S3 deaktiviert (veraltet). Danach akzeptiert Amazon S3 nur noch API-Anforderungen, die mithilfe der Signature Version 4 signiert wurden.

Dieser Abschnitt umfasst Antworten auf häufig gestellte Fragen im Bezug auf das Supportende für Signature Version 2.

Um was handelt es sich bei der Signature Version 2/4 und was versteht man unter dem Signieren einer Anforderung?

Der Signiervorgang mittels Signature Version 2 oder Signature Version 4 authentifiziert Ihre API-Anforderungen in Amazon S3. Durch das Signieren von Anforderungen ist Amazon S3 in der Lage, den Absender der Anforderung zu identifizieren und Ihre Anforderungen vor Angreifern zu schützen.

Weitere Informationen zum Signieren von AWS-Anforderungen finden Sie unter Signieren von AWS-API-Anforderungen in der Allgemeine AWS-Referenz.

Welche Aktualisierung nehmen Sie vor?

Derzeit unterstützen wir Amazon S3 API-Anforderungen, die mithilfe von Signature Version 2 und Signature Version 4 signiert wurden. Später akzeptiert Amazon S3 nur noch Anforderungen, die mit Signature Version 4 signiert wurden.

Weitere Informationen zum Signieren von AWS-Anforderungen finden Sie unter Änderungen in Signature Version 4 in der Allgemeine AWS-Referenz.

Warum nehmen Sie die Aktualisierung vor?

Signature Version 4 bietet mehr Sicherheit, da hierbei im Gegensatz zum geheimen Zugriffsschlüssel ein Signaturschlüssel verwendet wird. Signature Version 4 wird derzeit in allen AWS-Regionen unterstützt, während Signature Version 2 nur von Regionen unterstützt wird, die vor dem Januar 2014 gestartet wurden. Dank dieser Aktualisierung sind wir in der Lage, in allen Regionen eine einheitlichere Erfahrung bereitzustellen.

Wie kann ich sicherstellen, dass ich Signature Version 4 verwende und welche Aktualisierungen benötige ich?

Normalerweise entscheidet das Tool oder das SDK auf Clientseite, welche Signature Version zum Signieren der Anforderungen verwendet wird. Standardmäßig verwenden die aktuellen Versionen unsere AWS-SDKs Signature Version 4. Setzen Sie sich mit dem entsprechenden Supportteam für Ihre Drittanbietersoftware in Verbindung, um zu erfahren, welche Version Sie benötigen. Um direkte REST-Aufrufe an Amazon S3 zu senden, müssen Sie Ihre Anwendung so anpassen, dass der Signiervorgang mithilfe von Signature Version 4 ausgeführt wird.

Weitere Informationen darüber, welche Version des AWS-SDKs beim Wechsel zu Signature Version 4 zu verwenden ist, finden Sie unter Von Signature Version# 2 zur Signature Version 4 wechseln..

Weitere Informationen zur Verwendung von Signature Version 4 mit der Amazon S3 REST-API finden Sie unter Authenticating Requests (Authentifizieren von Anforderungen) (AWS Signature Version 4) in der API-Referenz zu Amazon Simple Storage Service.

Was passiert, wenn ich keine Aktualisierung durchführe?

Anforderungen, mit Signature Version 2 signiert wurden und zu einem späteren Zeitpunkt gesendet werden, können dann nicht mehr mit Amazon S3 authentifiziert werden. Anforderer sehen eine Fehlermeldung, dass die Anforderung mit Signature Version 4 signiert sein muss.

Sollte ich Änderungen vornehmen, selbst wenn ich eine vorsignierte URL verwende, die es erforderlich macht, dass ich für mehr als sieben Tage signiere?

Wenn sie eine vorsignierte URL verwenden, für die Sie für mehr als sieben Tage signieren müssen, ist keine Maßnahme erforderlich. Sie können weiterhin AWS Signature Version 2 zum Signieren und

Authentifizieren der vorsignierten URL verwenden. Wir werden uns darum kümmern und zusätzliche Einzelheiten zur Migration zu Signature Version 4 für ein vorab signiertes URL-Szenario bereitstellen.

Weitere Infos

- Weitere Informationen zur Verwendung von Signature Version 4 finden Sie unter Signing AWS API Requests (Signieren von AWS-API-Anforderungen).
- Sehen Sie sich die Änderungen zwischen Signature Version 2 und Signature Version 4 unter Änderungen in Signature Version 4 an.
- Lesen Sie den Beitrag AWS Signature Version 4 to replace AWS Signature Version 2 for signing Amazon S3 API requests in den AWS-Foren.
- Wenden Sie sich bei weiteren Fragen oder Bedenken unter AWS Support an uns.

Von Signature Version# 2 zur Signature Version 4 wechseln.

Wenn Sie aktuell Signature Version 2 für die Authentifizierung der API-Anforderungen in Amazon S3 verwenden, sollten Sie möglichst bald auf Signature Version 4 umwechseln. Das Ende der Unterstützung für Signature Version 2 wird unter beschriebe AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet).

Weitere Informationen zur Verwendung von Signature Version 4 mit der Amazon S3 REST-API finden Sie unter Authenticating Requests (Authentifizieren von Anforderungen) (AWS Signature Version 4) in der API-Referenz zu Amazon Simple Storage Service.

Die folgende Liste stellt die SDKs mit der erforderlichen Mindestversion zur Verwendung von Signature Version 4 (SigV4) dar. Wenn Sie vorsignierte URLs mit den AWS SDKs Java, JavaScript (Node.js) oder Python (Boto/CLI) verwenden SDKs, müssen Sie die richtige AWS-Region und Signature Version 4 in der Client-Konfiguration festlegen. Weitere Informationen zum Einrichten von `SigV4` in der Client-Konfiguration finden Sie unter Angabe der Signature-Version in der Anforderungs-Authentifizierung.

Wenn Sie diese SDK/ dieses Produkt verwenden	Aktualisieren Sie auf diese SDK-Version	Bei Verwendung von Sigv4 Codeänderung am Client erforderlich?	Link zur SDK-Dokumentation
AWS SDK for Java v1	Upgrade auf Java 1.11.201+ oder v2.	Ja	Angabe der Signature-Version in der Anforderungs-Authentifizierung
AWS SDK for Java v2	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK for Java
AWS SDK for .NET v1	Auf 3.1.10 oder höher aktualisieren.	Ja	AWS SDK for .NET
AWS SDK for .NET v2	Auf 3.1.10 oder höher aktualisieren.	Nein	AWS SDK for .NET v2
AWS SDK for .NET v3	Auf 3.3.0.0 oder höher aktualisieren.	Ja	AWS SDK for .NET v3
AWS SDK for JavaScript v1	Auf 2.68.0 oder höher aktualisieren.	Ja	AWS SDK for JavaScript
AWS SDK for JavaScript v2	Auf 2.68.0 oder höher aktualisieren.	Ja	AWS SDK for JavaScript

Wenn Sie diese SDK/ dieses Produkt verwenden	Aktualisieren Sie auf diese SDK-Version	Bei Verwendung von Sigv4 Codeänderung am Client erforderlich?	Link zur SDK-Dokumentation
AWS SDK for JavaScript v3	Derzeit ist keine weitere Maßnahme erforderlich. Im Q3 2019 auf Hauptversion V3 aktualisieren.	Nein	AWS SDK for JavaScript
AWS SDK for PHP v1	Empfohlen wird ein Upgrade auf die aktuelle Version von PHP oder mindestens v2.7.4 und der Wert v4 für den Signature-Parameter in der Konfiguration des S3-Clients.	Ja	AWS SDK for PHP

Wenn Sie diese SDK/ dieses Produkt verwenden	Aktualisieren Sie auf diese SDK-Version	Bei Verwendung von Sigv4 Codeänderung am Client erforderlich?	Link zur SDK-Dokumentation
AWS SDK for PHP v2	Empfohlen wird ein Upgrade auf die aktuelle Version von PHP oder mindestens v2.7.4 und der Wert v4 für den Signature-Parameter in der Konfiguration des S3-Clients.	Nein	AWS SDK for PHP
AWS SDK for PHP v3	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK for PHP
Boto2	Auf Boto2 v2.49.0. aktualisieren.	Ja	Boto 2-Aktualisierung
Boto3	Auf 1.5.71 (Botocore), 1.4.6 (Boto3) aktualisieren.	Ja	Boto 3 - AWS-SDK für Python

Wenn Sie diese SDK/ dieses Produkt verwenden	Aktualisieren Sie auf diese SDK-Version	Bei Verwendung von Sigv4 Codeänderung am Client erforderlich?	Link zur SDK-Dokumentation
AWS CLI	Auf 1.11.108 aktualisieren.	Ja	AWS Command Line Interface
AWS CLI v2 (Vorversion)	Keine SDK-Aktualisierung erforderlich.	Nein	AWS Command Line Interface Version 2
AWS SDK for Ruby v1	Auf Ruby V3 aktualisieren.	Ja	Ruby V3 für AWS
AWS SDK for Ruby v2	Auf Ruby V3 aktualisieren.	Ja	Ruby V3 für AWS
AWS SDK for Ruby v3	Keine SDK-Aktualisierung erforderlich.	Nein	Ruby V3 für AWS
Go	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK for Go
C++	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK for C++

AWS Tools for Windows PowerShell oder AWS Tools for PowerShell Core

Wenn Sie Modulversionen verwenden, die älter als 3.3.0.0 sind, müssen Sie auf 3.3.0.0 aktualisieren.

Verwenden Sie `Get-Module` cmdlet, um die Versionsinformationen zu erhalten:

```
Get-Module -Name AWSPowershell
Get-Module -Name AWSPowershell.NetCore
```

Um auf die Version 3.3.0.0 zu aktualisieren, verwenden Sie das cmdlet `Update-Module`:

```
Update-Module -Name AWSPowershell
Update-Module -Name AWSPowershell.NetCore
```

Für Signature Version 2-Datenverkehr können Sie vorab signierte URLs verwenden, die länger als sieben Tage gültig sind.

Verwendung der AWS SDK for Java

Das AWS SDK for Java stellt eine API für den Amazon-S3-Bucket und Objektoperationen bereit. Für Objektoperationen stellt das SDK – zusätzlich zu der API für das Hochladen von Objekten in einer einzigen Operation – eine API zum mehrteiligen Hochladen großer Objekte bereit. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

Themen

- Die Java API-Organisation
- Testen der Java-Codebeispiele für Amazon S3

Das AWS SDK for Java bietet Ihnen die Option, eine High-Level- oder eine Low-Level-API zu verwenden.

API auf niedriger Ebene

Die Low-Level-APIs entsprechen den zugrunde liegenden Amazon-S3-REST-Operationen, wie beispielsweise zum Erstellen, Aktualisieren und Löschen, die auf Buckets und Objekte angewendet werden. Wenn Sie große Objekte über die Low-Level-API für mehrteilige Uploads hochladen, bietet dies eine bessere Kontrolle. Beispielsweise können Sie damit mehrteilige Uploads unterbrechen und

fortsetzen, die Teilegrößen während des Uploads ändern oder Uploads beginnen, obwohl Sie die Größe der Daten nicht vorab kennen. Wenn Sie diese Anforderungen nicht haben, verwenden Sie die High-Level API zum Hochladen von Objekten.

API auf hoher Ebene

Zum Hochladen von Objekten bietet das SDK einen höheren Abstraktionsgrad, indem es die Klasse `TransferManager` bereitstellt. Die High-Level API ist eine einfachere API, wobei Sie mit nur ein paar Codezeilen Dateien und Streams in Amazon S3 hochladen können. Sie sollten diese API verwenden, um Daten hochzuladen, es sei denn, Sie müssen den Upload kontrollieren, wie im vorigen Abschnitt über die Low-Level API beschrieben.

Für kleinere Datengrößen lädt die `TransferManager`-API Daten in einer einzigen Operation hoch. Der `TransferManager` wechselt jedoch zur Verwendung der API für den mehrteiligen Upload, wenn die Datengröße eine bestimmte Schwelle erreicht. Wenn möglich, verwendet der `TransferManager` mehrere Threads, um die Teile nebenläufig hochzuladen. Wenn der Upload eines Teils fehlschlägt, wiederholt API diesen Upload bis zu dreimal. Dies sind jedoch Optionen, die mit der `TransferManagerConfiguration`-Klasse konfiguriert werden können.

> **Note**
>
> Wenn Sie einen Stream für die Datenquelle verwenden, führt die `TransferManager`-Klasse keine gleichzeitigen Uploads durch.

Die Java API-Organisation

Die folgenden Pakete in AWS SDK for Java stellen die API bereit:

- com.amazonaws.services.s3 – Stellt die APIs für das Erstellen von Amazon-S3-Clients und für das Arbeiten mit Buckets und Objekten bereit. Dieses Paket ermöglicht Ihnen beispielsweise, Buckets zu erstellen, Objekte hochzuladen, Objekte abzurufen, Objekte zu löschen und Schlüssel aufzulisten.
- com.amazonaws.services.s3.transfer – Stellt die High-Level-API-Datenoperationen bereit.

  Diese High-Level-API ist darauf ausgelegt, das Übertragen von Objekten in und aus Amazon S3 zu vereinfachen. Dieses Paket beinhaltet die `TransferManager`-Klasse, die asynchrone Methoden für das Arbeiten mit sowie Abfragen und Bearbeiten von Übertragungen bereitstellt. Außerdem umfasst sie die `TransferManagerConfiguration`-Klasse, mit der Sie die Mindestteilgröße

für den mehrteiligen Upload konfigurieren können, sowie den Schwellenwert (in Bytes), ab wann mehrteilige Uploads verwendet werden sollen.

- com.amazonaws.services.s3.model – Stellt die Low-Level API-Klassen zum Erstellen von Anfragen und zum Verarbeiten von Antworten bereit. Es beinhaltet beispielsweise die Klasse `GetObjectRequest`, um Ihre "Get object"-Anforderung zu beschreiben, die Klasse `ListObjectsRequest`, um Ihre Listenschlüsselanforderungen zu beschreiben, und die Klasse `InitiateMultipartUploadRequest`, um mehrteilige Uploads zu erstellen.

Weitere Informationen zur AWS SDK for Java-API finden Sie in der AWS SDK for Java-API-Referenz.

Testen der Java-Codebeispiele für Amazon S3

Die Java-Beispiele in diesem Handbuch sind sämtlich mit der AWS SDK for Java-Version 1.11.321 kompatibel. Für Anleitungen zum Einrichten und Ausführen von Codebeispielen vgl. Getting Started (Erste Schritte) im AWS SDK for Java-Entwicklerhandbuch.

Verwendung der AWS SDK for .NET

Das AWS SDK for .NET stellt eine API für den Amazon-S3-Bucket und Objektoperationen bereit. Für Objektoperationen bietet der SDK zusätzlich zu der API für das Hochladen von Objekten in einer einzigen Operation die API zum mehrteiligen Hochladen großer Objekte bereit (siehe Hochladen und Kopieren von Objekten mit mehrteiligen Uploads).

Themen

- Die .NET API Organisation
- Ausführen der .NET-Codebeispiele für Amazon S3

Das AWS SDK for .NET bietet Ihnen die Option, eine High-Level- oder eine Low-Level-API zu verwenden.

API auf niedriger Ebene

Die Low-Level-APIs entsprechen den zugrunde liegenden Amazon-S3-REST-Vorgängen, wie beispielsweise zum Erstellen, Aktualisieren und Löschen, die auf Buckets und Objekte angewendet werden. Wenn Sie große Objekte über die Low-Level-API für mehrteilige Uploads hochladen (siehe Hochladen und Kopieren von Objekten mit mehrteiligen Uploads), bietet dies eine bessere Kontrolle. Beispielsweise können Sie damit mehrteilige Uploads unterbrechen und fortsetzen, die Teilegrößen während des Uploads ändern oder Uploads beginnen, obwohl Sie die Größe der Daten nicht vorab

kennen. Wenn Sie diese Anforderungen nicht haben, verwenden Sie die High-Level API zum Hochladen von Objekten.

API auf hoher Ebene

Zum Hochladen von Objekten bietet das SDK einen höheren Abstraktionsgrad, indem es die Klasse `TransferUtility` bereitstellt. Die High-Level API ist eine einfachere API, wobei Sie mit nur ein paar Codezeilen Dateien und Streams in Amazon S3 hochladen können. Sie sollten diese API verwenden, um Daten hochzuladen, es sei denn, Sie müssen den Upload kontrollieren, wie im vorigen Abschnitt über die Low-Level API beschrieben.

Für kleinere Datengrößen lädt die `TransferUtility`-API Daten in einer einzigen Operation hoch. Der `TransferUtility` wechselt jedoch zur Verwendung der API für den mehrteiligen Upload, wenn die Datengröße eine bestimmte Schwelle erreicht. Standardmäßig verwendet er mehrere Threads, um die Teile nebenläufig hochzuladen. Wenn der Upload eines Teils fehlschlägt, wiederholt API diesen Upload bis zu dreimal. Dies sind jedoch konfigurierbare Optionen.

> **Note**
>
> Wenn Sie einen Stream für die Datenquelle verwenden, führt die `TransferUtility`-Klasse keine gleichzeitigen Uploads durch.

Die .NET API Organisation

Wenn Sie Amazon-S3-Anwendungen mit dem AWS SDK for .NET schreiben, verwenden Sie die `AWSSDK.dll`. Die folgenden Namespaces in dieser Gruppe stellen die API für mehrteilige Uploads bereit:

- Amazon.S3.Transfer – Stellt die High-Level API für den mehrteiligen Upload Ihrer Daten bereit.

  Beinhaltet die `TransferUtility`-Klasse, die Ihnen ermöglicht, eine Datei, ein Verzeichnis oder einen Stream zum Hochladen Ihrer Daten bereitzustellen. Darüber hinaus beinhaltet sie die Klassen `TransferUtilityUploadRequest` und `TransferUtilityUploadDirectoryRequest`, um erweiterte Einstellungen zu konfigurieren, wie beispielsweise die Anzahl gleichzeitiger Threads, die Teilegröße, die Objekt-Metadaten, die Speicherklasse (STANDARD, REDUCED_REDUNDANCY) und die Objekt-Zugriffskontrollliste (ACL).
- Amazon.S3 – Stellt die Implementierung für Low-Level APIs bereit.

Bietet Methoden, die der Amazon-S3-REST-API für den mehrteiligen Upload entsprechen (siehe Verwenden der REST-API).

- Amazon.S3.Model –Stellt die Low-Level API-Klassen zum Erstellen von Anfragen und zum Verarbeiten von Antworten bereit. Sie beinhaltet beispielsweise die Klassen `InitiateMultipartUploadRequest` und `InitiateMultipartUploadResponse`, die Sie für die Initiierung eines mehrteiligen Uploads verwenden können, und die Klassen `UploadPartRequest` und `UploadPartResponse`, wenn Teile hochgeladen werden.
- Amazon.S3.Encryption – Stellt `AmazonS3EncryptionClient` bereit.
- Amazon.S3.Util – Stellt verschiedene Klassen wie z. B. `AmazonS3Util` und `BucketRegionDetector` bereit.

Weitere Informationen zur AWS SDK for .NET-API finden Sie unter AWS-SDK für .NET Version 3 API-Referenz.

Ausführen der .NET-Codebeispiele für Amazon S3

Die .NET-Codebeispiele in diesem Handbuch sind sämtlich mit der AWS SDK for .NET-Version 3.0 kompatibel. Informationen zum Einrichten und Ausführen der Codebeispiele finden Sie unter Getting started with the AWS SDK for .NET (Erste Schritte mit dem AWS-SDK für .NET) im Entwicklerbuch für AWS-SDK für .NET.

Verwenden von AWS SDK for PHP und Ausführen von PHP-Beispielen

Das AWS SDK for PHP bietet Zugriff auf die API für Amazon-S3-Bucket- und Objektoperationen. Der SDK bietet Ihnen die Option, die High-Level-API oder höhere Abstraktionen des Service zu verwenden.

Das SDK steht unter AWS SDK for PHP zur Verfügung und enthält auch Anweisungen für die Installation und die ersten Schritte mit dem SDK.

Die Einrichtung für die Verwendung von AWS SDK for PHP ist von Ihrer Umgebung abhängig, und wie Sie Ihre Anwendung ausführen wollen. Informationen zum Einrichten Ihrer Umgebung für die Ausführung der Beispiele in dieser Dokumentation finden Sie im Handbuch „Erste Schritte“ zum AWS-SDK für PHP.

Themen

- AWS SDK for PHP-Ebenen

- PHP-Beispiele ausführen
- Verwandte Ressourcen

AWS SDK for PHP-Ebenen

Das AWS SDK for PHP bietet Ihnen die Option, eine High-Level- oder eine Low-Level-API zu verwenden.

API auf niedriger Ebene

Die Low-Level-APIs entsprechen den zugrunde liegenden Amazon-S3-REST-Vorgängen, wie beispielsweise zum Erstellen, Aktualisieren und Löschen, die auf Buckets und Objekte angewendet werden. Die Low-Level-APIs bieten mehr Kontrolle über diese Operationen. Beispielsweise können Sie Ihre Anforderungen zusammenfassen und parallel ausführen. Bei Verwendung der API für mehrteilige Uploads können Sie die Objektteile unabhängig voneinander verwalten. Beachten Sie, dass diese Low-Level-API-Aufrufe ein Ergebnis zurückgeben, das alle Amazon-S3-Antwortdetails beinhaltet. Weitere Informationen zur API für mehrteilige Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.

High-Level-Abstraktionen

Die High-Level-Abstraktionen sind darauf ausgelegt, häufige Anwendungsfälle zu vereinfachen. Um beispielsweise große Objekte unter Verwendung der Low-Level-API hochzuladen, müssen Sie zuerst `Aws\S3\S3Client::createMultipartUpload()` aufrufen, dann die Methode `Aws\S3\S3Client::uploadPart()`, um die Objektteile hochzuladen, und dann die Methode `Aws\S3\S3Client::completeMultipartUpload()`, um den Upload abzuschließen. Sie können statt dessen das `Aws\S3\\MultipartUploader`-Objekt verwenden, das sich auf einer höheren Ebene befindet, und das die Erstellung mehrteiliger Uploads vereinfacht.

Als weiteres Beispiel für die Auflistung von Objekten in einem Bucket können Sie die Iterator-Funktion von AWS SDK for PHP verwenden, um alle Objektschlüssel zurückzugeben, unabhängig davon, wie viele Objekte Sie in dem Bucket gespeichert haben. Wenn Sie die Low-Level-API verwenden, gibt die Antwort maximal 1000 Schlüssel zurück. Wenn ein Bucket mehr als 1000 Objekte enthält, wird das Ergebnis abgeschnitten. Sie müssen die Antwort verwalten und auf Kürzungen überprüfen.

PHP-Beispiele ausführen

Informationen zum Einrichten und Verwenden der Amazon-S3-Beispiele für Version 3 des AWS-SDK für PHP finden Sie unter Installation im AWS SDK for PHP-Entwicklerhandbuch.

Verwandte Ressourcen

- AWS SDK for PHP für Amazon S3
- AWS-SDK für PHP Dokumentation
- AWS-SDK für PHP-API für Amazon S3
- Code-Beispiele für AWS-SDK for PHP Version 3

Verwenden von AWS SDK for Ruby – Version 3

Das AWS SDK for Ruby stellt eine API für Amazon-S3-Bucket und Objektoperationen bereit. Für Objektoperationen können Sie die API für das Hochladen von Objekten in einer einzigen Operation oder die API zum mehrteiligen Hochladen großer Objekte verwenden (siehe Hochladen eines Objekts mit Multipart-Upload). Die API für den Upload in einer Operation kann jedoch keine großen Objekte verarbeiten, und hinter den Kulissen wird der mehrteilige Upload für Sie verwaltet. Damit wird die Anzahl der Skripts reduziert, die Sie schreiben müssen.

Die Ruby API-Organisation

Wenn Sie Amazon-S3-Anwendungen mit dem AWS SDK for Ruby-SDK für Ruby erstellen, müssen Sie das SDK-für-Ruby-Gem installieren. Weitere Informationen finden Sie unter AWS-SDK für Ruby – Version 3. Nachdem Sie es installiert haben, können Sie auf die API zugreifen, einschließlich der folgenden Schlüsselklassen:

- Aws::S3::Resource –Stellt die Schnittstelle zu Amazon S3 für das Ruby SDK dar und bietet Methoden zum Erstellen und Auflisten von Buckets.

  Die Klasse`S3` stellt die Instance-Methode `#buckets` für den Zugriff auf vorhandene Buckets oder zum Erstellen neuer Buckets bereit.
- Aws::S3::Bucket – Stellt einen Amazon-S3-Bucket dar.

  Die Klasse `Bucket` stellt die Methoden `#object(key)` und `#objects` für den Zugriff auf die Objekte in einem Bucket bereit, ebenso wie die Methoden zum Löschen eines Buckets und zur Rückgabe von Informationen über einen Bucket, wie beispielsweise die Bucket-Richtlinie.
- Aws::S3::Object – Stellt ein durch seinen Schlüssel identifiziertes Amazon-S3-Objekt dar.

  Die `Object`-Klasse bietet Methoden für den Abruf und die Einrichtung der Eigenschaften eines Objekts, mit Angabe der Speicherklasse zum Speichern von Objekten, und der Einrichtung

von Objektberechtigungen unter Verwendung von Zugriffskontrolllisten. Die Klasse `Object` enthält auch Methoden zum Löschen, Hochladen und Kopieren von Objekten. Beim mehrteiligen Hochladen von Objekten stellt Ihnen diese Klasse Optionen bereit, um die Reihenfolge der hochgeladenen Teile sowie die Teilegröße anzugeben.

Weitere Informationen zur AWS-SDK für Ruby API finden Sie unter AWS-SDK für Ruby – Version 2.

Testen der Ruby Script-Codebeispiele

Am einfachsten gelingt der Einstieg in die Ruby Script--Codebeispiele, indem das neuste AWS SDK for Ruby Gem installiert wird. Weitere Informationen zur Installation und Aktualisierung der neuesten Gem finden Sie unter AWS-SDK für Ruby – Version 3. Die folgenden Aufgaben führen Sie durch das Erstellen und Testen der Ruby Script-Beispiele, wobei vorausgesetzt wird, dass Sie installiert habe AWS SDK for Ruby.

Allgemeiner Vorgang beim Erstellen und Testen von Ruby-Skriptbeispielen

1	Für den Zugriff auf AWS müssen Sie die Anmeldeinformationen für Ihre SDK für Ruby-Anwendung bereitstellen. Weitere Informationen finden Sie unter Konfigurieren des AWS-SDK für Ruby.
2	Erstellen Sie ein neues Skript für SDK for Ruby und fügen Sie oben die folgenden Codezeilen ein. `#!/usr/bin/env ruby` `require 'rubygems'` `require 'aws-sdk-s3'` Die erste Zeile ist die Interpreter-Anweisung, und die beiden `require`-Anweisungen importieren zwei erforderliche Gems in Ihr Skript.
3	Kopieren Sie das Code aus dem Abschnitt, den Sie gerade lesen, in Ihr Skript.
4	Aktualisieren Sie den Code mit den erforderlichen Daten. Wenn Sie beispielsweise eine Datei hochladen, geben Sie den Dateipfad und den Bucket-Namen an.

5	Führen Sie das Skript aus. Überprüfen Sie Änderungen an Buckets und Objekten unter Verwendung der AWS Management Console. Weitere Informationen zum AWS Management Console finden Sie unter https://aws.amazon.com/console/.

Ruby-Beispiele

Die folgenden Links enthalten Beispiele, die Ihnen bei den ersten Schritten mit dem SDK for Ruby – Version 3 helfen:

- Erstellen eines Buckets
- Objekte hochladen

Verwendung der AWS SDK for Python (Boto)

Boto ist ein Python-Paket, das Schnittstellen für AWS bereitstellt, einschließlich Amazon S3. Weitere Informationen zu Boto finden Sie unter AWS SDK for Python (Boto). Der Link „Erste Schritte“ auf dieser Seite enthält step-by-step Anweisungen für die ersten Schritte.

Verwenden der AWS-Mobile-SDKs für iOS und Android

Mit den AWS-Mobile-SDKs für Android und iOS können Sie robuste Cloud-Backends schnell und einfach in Ihre vorhandenen mobilen Apps integrieren. Sie können Funktionen wie Benutzeranmeldung, Datenbanken, Push-Benachrichtigungen und vieles mehr konfigurieren und nutzen, ohne ein AWS-Experte zu sein.

Die AWS-Mobile-SDKs bieten einfachen Zugriff auf Amazon S3 und viele andere AWS-Services. Informationen zum Einstieg in die Arbeit mit AWS-Mobile-SDKs finden Sie unter Erste Schritte mit den AWS-Mobile-SDKs.

Weitere Infos

Verwenden der AWS Amplify JavaScript Library

Verwenden der AWS Amplify JavaScript Library

AWS Amplify ist eine Open-Source- JavaScript Bibliothek für Web- und Mobilentwickler, die Cloud-fähige Anwendungen erstellen. AWS Amplify bietet anpassbare UI-Komponenten und eine

deklarative Schnittstelle für die Verwendung mit einem S3-Bucket, zusammen mit anderen High-Level-Kategorien für AWS-Services.

Um mit der Verwendung der AWS Amplify- JavaScript Bibliothek zu beginnen, wählen Sie einen der folgenden Links aus:

- Erste Schritte mit der AWS-Amplify-Bibliothek für Web
- Erste Schritte mit Amplify

Weitere Informationen zu AWS Amplify finden Sie unter AWS Amplify auf GitHub.

Weitere Infos

Verwenden der AWS-Mobile-SDKs für iOS und Android

Verwendung der AWS SDK for JavaScript

stellt eine JavaScript API für -AWSServices AWS SDK for JavaScript bereit. Sie können die JavaScript -API verwenden, um Bibliotheken oder Anwendungen für Node.js oder den Browser zu erstellen.

Weitere Informationen zur Verwendung des AWS SDK for JavaScript für JavaScript für Amazon S3 finden Sie unten.

- Was ist AWS SDK for JavaScript? (v2)
- AWS SDK for JavaScript – Amazon-S3-Beispiele (v2)
- AWS SDK for JavaScript API-Referenz für Amazon S3 (v2)
- Was ist AWS SDK for JavaScript? (v3)
- AWS SDK for JavaScript – Amazon-S3-Beispiele (v3)
- AWS SDK for JavaScript API-Referenz für Amazon S3 (v3)

Entwickeln mit Amazon S3 unter Verwendung der REST-API

Die Amazon-S3-Architektur ist so ausgelegt, dass sie unabhängig von Programmiersprachen ist und unsere unterstützten Schnittstellen verwendet, um Objekte zu speichern und abzurufen.

Amazon S3 bietet derzeit eine REST-Schnittstelle. Mit REST werden Metadaten in HTTP-Headern zurückgegeben. Wir unterstützen nur HTTP-Anfragen von bis zu 4 KB (ohne den Rumpf), deshalb

ist die Menge der von Ihnen bereitgestellten Metadaten begrenzt. Die REST-API ist eine HTTP-Schnittstelle zu Amazon S3. Mit REST verwenden Sie HTTP-Standardanfragen, um Buckets und Objekte zu erstellen, laden oder löschen.

Sie können einen beliebigen Toolkit einsetzen, der HTTP unterstützt, um die REST-API verwenden zu können. Sie können sogar einen Browser verwenden, um Objekte zu laden, wenn diese anonym lesbar sind.

Die REST-API verwendet die HTTP-Standardheader und -Statuscodes, sodass sich Standard-Browser und -Toolkits wie erwartet verhalten. In einigen Bereichen haben wir HTTP um zusätzliche Funktionen erweitert (wir haben beispielsweise Header hinzugefügt, um die Zugriffskontrolle zu unterstützen). In diesen Fällen haben wir alles dafür getan, die neue Funktion so hinzuzufügen, dass sie der standardmäßigen Nutzung von HTTP entsprechen.

Weitere Informationen zum Senden von Anforderungen mit der REST-API finden Sie unter Senden von Anforderungen unter Verwendung der REST-API. In den folgenden Themen finden Sie einige Überlegungen, die Sie bei der Verwendung der REST API beachten sollten.

Weitere Informationen zur REST-API von Amazon S3 finden Sie in der Amazon Simple Storage Service API-Referenz.

Themen

- Weiterleitung von Anforderungen

Weiterleitung von Anforderungen

Programme, die Anfragen für Buckets stellen, die mit der CreateBucket-API und einer CreateBucketConfiguration erstellt wurden, müssen Umleitungen unterstützen. Darüber hinaus entstehen möglicherweise für einige Clients Probleme, die DNS TTLs nicht unterstützen.

Dieser Abschnitt beschreibt Weiterleitungs- und DNS-Aspekte, die Sie beim Entwurf Ihres Service oder Ihrer Anwendung für Amazon S3 berücksichtigen sollten.

Anforderungsumleitung und die REST-API

Amazon S3 verwendet das DNS (Domain Name System), um Anfragen an Einrichtungen weiterzuleiten, die sie verarbeiten können. Dieses System arbeitet effektiv, aber es können temporäre Routing-Fehler auftreten. Trifft eine Anfrage am falschen Amazon-S3-Standort ein, reagiert Amazon

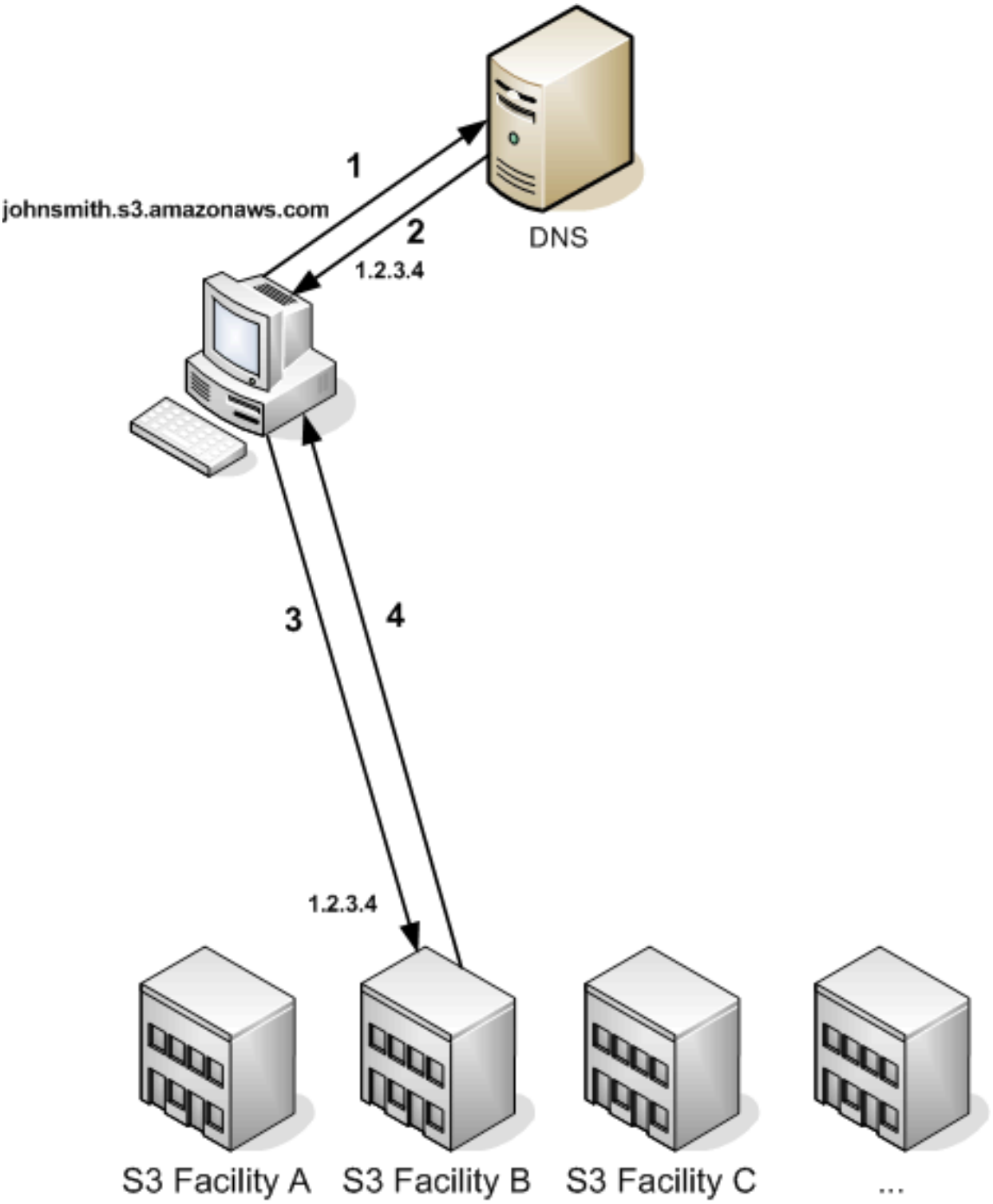
S3 mit einer temporären Umleitung, die den Auftraggeber anweist, die Anfrage an einen neuen Endpunkt zu senden. Ist eine Anfrage fehlerhaft aufgebaut, verwendet Amazon S3 permanente Umleitungen, um dem Auftraggeber mitzuteilen, wie die Anfrage korrekt ausgeführt wird.

> **Important**
>
> Um diese Funktion nutzen zu können, benötigen Sie eine Anwendung, die Amazon-S3-Umleitungsantworten verarbeiten kann. Die einzige Ausnahme bilden Anwendungen, die ausschließlich mit Buckets arbeiten, die ohne erstellt wurde `<CreateBucketConfiguration>`. Weitere Informationen über Standorteinschränkungen finden Sie unter Zugreifen auf einen Amazon-S3-Bucket und Auflisten des Buckets.
> Für alle Regionen, die nach dem 20. März 2019 gestartet wurden, gibt Amazon S3 den Fehler „HTTP 400 Bad Request“ zurück, wenn eine Anforderung an einer falschen Amazon-S3-Position empfangen wird.
> Weitere Informationen zum Aktivieren oder Deaktivieren einer AWS-Region finden Sie unter AWS-Regionen und -Endpunkte in der Allgemeine AWS-Referenz.

Themen

DNS-Weiterleitung

Die DNS-Weiterleitung leitet Anfragen an geeignete Amazon-S3-Einrichtungen weiter. Die folgende Abbildung und Vorgehensweise zeigen ein Beispiel für DNS-Routing.



DNS-Routing-Anforderungsschritte

1. Der Client stellt eine DNS-Anfrage, um ein Objekt in Amazon S3 speichern zu lassen.

2. Der Client erhält eine oder mehrere IP-Adressen für Einrichtungen, die die Anfrage verarbeiten können. In diesem Beispiel ist die IP-Adresse für die Einrichtung B.
3. Der Client stellt eine Anforderung an Amazon-S3-Einrichtung B.
4. Einrichtung B gibt eine Kopie des Objekts an den Kunden zurück.

Temporäre Anforderungsumleitung

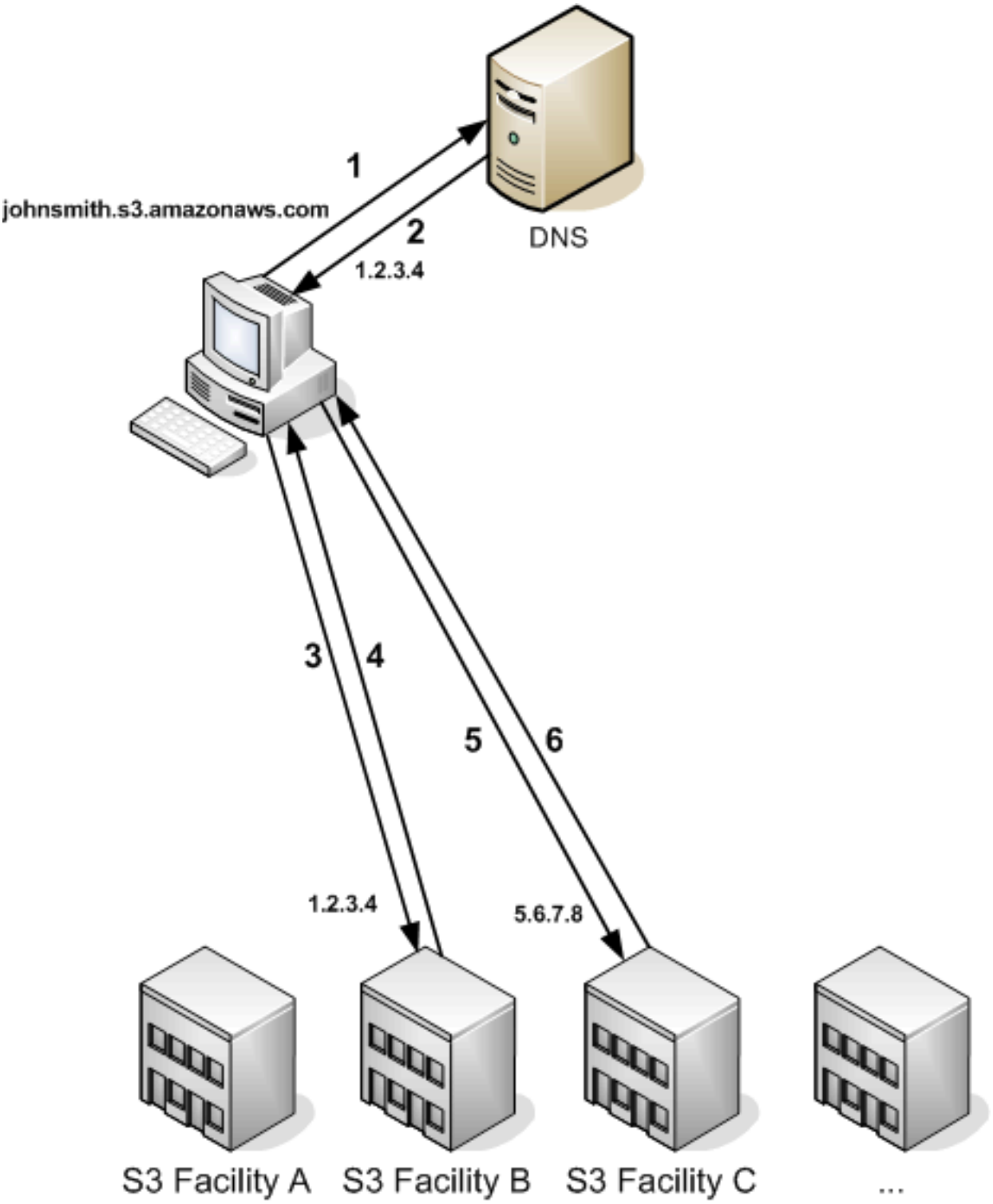
Eine temporäre Umleitung ist eine Art Fehlermeldung, die dem Auftraggeber signalisiert, dass er die Anfrage an einen anderen Endpunkt senden sollte. Aufgrund der verteilten Natur von Amazon S3 können Anfragen temporär in die falsche Einrichtung weitergeleitet werden. Dies tritt am wahrscheinlichsten auf, unmittelbar nachdem Buckets erstellt oder gelöscht wurden.

Wenn Sie beispielsweise einen neuen Bucket erstellen und sofort eine Anfrage für den Bucket machen, werden Sie möglicherweise temporär umgeleitet, abhängig von der Standorteinschränkung des Buckets. Wenn Sie den Bucket in der AWS-Region USA Ost (Nord-Virginia) erstellt haben, wird die Umleitung nicht angezeigt, da dies auch der Standard-Amazon-S3-Endpunkt ist.

Wenn der Bucket jedoch in einer anderen Region erstellt wird, gehen alle Anfragen für den Bucket an den Standard-Endpunkt, während der DNS-Eintrag des Buckets propagiert wird. Der Standard-Endpunkt leitet die Anfrage mit einer HTTP 302-Antwort an den richtigen Endpunkt weiter. Temporäre Umleitungen enthalten eine URI zur richtigen Einrichtung, die Sie verwenden können, um die Anfrage sofort erneut zu senden.

> **Important**
>
> Verwenden Sie keinen Endpunkt wieder, der aus einer vorherigen Umleitungsantwort stammt. Das scheint zu funktionieren (manchmal sogar sehr lange), kann aber unvorhersehbare Ergebnisse erzeugen und schlägt irgendwann ohne weitere Mitteilung fehl.

Die folgende Abbildung und Vorgehensweise zeigen ein Beispiel für eine temporäre Umleitung.



Temporäre Umleitungsschritte für Anfragen

1. Der Client stellt eine DNS-Anfrage, um ein Objekt in Amazon S3 speichern zu lassen.

2. Der Client erhält eine oder mehrere IP-Adressen für Einrichtungen, die die Anfrage verarbeiten können.
3. Der Client stellt eine Anforderung an Amazon-S3-Einrichtung B.
4. Einrichtung B gibt eine Umleitung zurück, die angibt, dass das Objekt an Standort C zur Verfügung steht.
5. Der Client sendet Anfrage noch einmal an Einrichtung C.
6. Einrichtung C gibt eine Kopie des Objekts zurück.

Permanente Anforderungsumleitung

Eine permanente Anfrageumleitung zeigt an, dass Ihre Anfrage eine fehlerhafte Adresse für eine Ressource angegeben hat. Permanente Umleitungen treten beispielsweise auf, wenn Sie eine Anfrage mit Angabe eines Pfads für den Zugriff auf einen Bucket verwenden, der mit erstellt wurd `<CreateBucketConfiguration>`. Weitere Informationen finden Sie unter Zugreifen auf einen Amazon-S3-Bucket und Auflisten des Buckets.

Um diese Fehler bei der Entwicklung zu erkennen, enthält diese Art Umleitung keinen HTTP Location-Header, der Ihnen ermöglicht, die Anfrage automatisch an den richtigen Standort zu verfolgen. Weitere Informationen über die Verwendung des richtigen Amazon-S3-Endpunkts finden Sie im resultierenden XML-Fehlerdokument.

Beispiele für Anforderungsumleitung

Im Folgenden finden Sie Beispiele für temporäre Umleitungsantworten.

Temporäre Umleitungsantwort der REST-API

```
HTTP/1.1 307 Temporary Redirect
Location: http://awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com/photos/puppy.jpg?
rk=e2c69a31
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Fri, 12 Oct 2007 01:12:56 GMT
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
```

```
  <Message>Please re-send this request to the specified temporary endpoint.
  Continue to use the original request endpoint for future requests.</Message>
  <Endpoint>awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com</Endpoint>
</Error>
```

Temporäre Umleitungsantwort der SOAP API

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.TemporaryRedirect</Faultcode>
    <Faultstring>Please re-send this request to the specified temporary endpoint.
    Continue to use the original request endpoint for future requests.</Faultstring>
    <Detail>
      <Bucket>images</Bucket>
      <Endpoint>s3-gztb4pa9sq.amazonaws.com</Endpoint>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

Überlegungen zu DNS

Eine der Entwurfsanforderungen von Amazon S3 ist eine extrem hohe Verfügbarkeit. Unter anderem erfüllen wir diese Anforderung, indem wir die dem Amazon-S3-Endpunkt in DNS zugeordnete IP-Adressen nach Bedarf aktualisieren. Diese Änderungen werden in kurzlebigen Clients automatisch reflektiert, nicht jedoch in einigen langlebigen Clients. Für langlebige Clients ist eine spezielle Maßnahme erforderlich, um den Amazon-S3-Endpunkt regelmäßig wieder aufzulösen, um von diesen Änderungen zu profitieren. Weitere Informationen zu virtuellen Maschinen (VMs) finden Sie hier:

- Für Java speichert die JVM von Sun die DNS-Suchen standardmäßig auf unbegrenzte Zeit. Informationen zum Ändern dieses Verhaltens finden Sie im Abschnitt „InetAddress Caching“ in der InetAddress-Dokumentation.

- Für PHP speichert die persistente PHP VM, die in den gebräuchlichsten Bereitstellungskonfigurationen ausgeführt wird, DNS-Suchen, bis die VM neu gestartet wird. Weitere Informationen finden Sie in den getHostByName-PHP-Dokumenten.

Behandlung von REST- und SOAP-Fehlern

Themen

Dieser Abschnitt beschreibt REST- und SOAP-Fehler und wie sie gehandhabt werden.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Die REST-Fehlerantwort

Wenn auf eine REST-Anfrage ein Fehler zurückgegeben wird, ist die HTTP-Antwort wie folgt aufgebaut:

- Ein XML-Fehlerdokument als Antworttext
- Content-Type: application/xml
- Der entsprechende 3xx-, 4xx- oder 5xx-HTTP-Statuscode

Nachfolgend finden Sie ein Beispiel für eine REST-Fehlermeldung.

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>NoSuchKey</Code>
  <Message>The resource you requested does not exist</Message>
```

```
  <Resource>/mybucket/myfoto.jpg</Resource>
  <RequestId>4442587FB7D0A2F9</RequestId>
</Error>
```

Weitere Informationen zu Amazon-S3-Fehlern finden Sie unter ErrorCodeList.

Antwort-Header

Die folgenden Antwort-Header werden von allen Operationen zurückgegeben:

- `x-amz-request-id:` Eine eindeutige ID, die das System jeder Anfrage zuordnet. In dem unwahrscheinlichen Fall, dass Sie Probleme mit Amazon S3 haben, kann Ihnen diese helfen, das Problem zu beheben.
- `x-amz-id-2:` Ein spezielles Token, das uns hilft, Probleme aufzulösen.

Antwort auf einen Fehler

Bei einer fehlerhaften Amazon-S3-Anforderung erhält der Client eine Fehlermeldung zurück. Das genaue Format der Fehlermeldung ist spezifisch für die API. Die REST-Fehlermeldung beispielsweise unterscheidet sich von der SOAP-Fehlermeldung. Alle Fehlermeldungen haben jedoch gemeinsame Elemente.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Fehlercode

Der Fehlercode ist eine Zeichenfolge handelt, die eine Fehlerbedingung eindeutig identifiziert. Er ist für Programme vorgesehen, die Fehler dem Typ nach erkennen und verarbeiten. Viele Fehlercode gibt es für SOAP- und REST-APIs, einige sind jedoch spezifisch für die API. NoSuchKey beispielsweise ist universell, während UnexpectedContent nur in Reaktion auf eine ungültige REST-Anfrage auftreten kann. In jedem Fall haben SOAP-Fehlercodes ein Präfix, das in der Fehlercodetabelle angegeben ist, ein NoSuchKey-Fehler wird also in SOAP als Client.NoSuchKey zurückgegeben.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Fehlermeldung

Die Fehlermeldung enthält eine generische Beschreibung der Fehlerbedingung in englischer Sprache. Sie ist für dafür vorgesehen, von Menschen gelesen zu werden. Einfache Programme zeigen dem Endbenutzer die Meldung direkt an, wenn sie auf eine Fehlerbedingung treffen, die sie nicht kennen, oder die sie nicht verarbeiten können. Komplexere Programme mit einer umfangreicheren Fehlerverarbeitung und einer geeigneten Internationalisierung werden die Fehlermeldung eher ignorieren.

Weitere Details

Viele Fehlermeldungen enthalten zusätzliche strukturierte Daten, die von einem Entwickler gelesen und interpretiert werden, der Programmfehler diagnostiziert. Wenn Sie beispielsweise einen Content-MD5-Header mit einer REST PUT-Anfrage senden, die nicht mit dem auf dem Server berechneten Digest übereinstimmt, erhalten Sie einen BadDigest-Fehler. Die Fehlermeldung enthält auch als Details den von uns berechneten Digest, sowie den Digest, den Sie erwarten und uns mitgeteilt haben. Während der Entwicklung können Sie diese Informationen nutzen, um den Fehler zu diagnostizieren. In der Produktion könnte ein ordnungsgemäß funktionierendes Programm diese Informationen in seinem Fehlerprotokoll enthalten.

Die SOAP-Fehlerantwort

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

In SOAP wird eine Fehlermeldung als SOAP-Fehler an den Client zurückgegeben, mit dem HTTP-Antwortcode 500. Wenn Sie keinen SOAP-Fehler erhalten, war Ihre Anfrage erfolgreich. Der Amazon S3SOAP-Fehlercode besteht aus einem SOAP 1.1-Standardfehlercode („Server“ oder „Client“),

verknüpft mit dem Amazon-S3-spezifischen Fehlercode. Beispiel: "Server.InternalError" oder "Client.NoSuchBucket". Das SOAP-Fehlerzeichenfolgenelement enthält eine generische, vom Menschen lesbare Fehlermeldung in englischer Sprache. Das SOAP-Fehlerdetailelement schließlich enthält verschiedene für den Fehler relevante Informationen.

Wenn Sie beispielsweise versuchen, das nicht existierende Objekt "Fred" zu löschen, enthält der Rumpf der SOAP-Antwort den SOAP-Fehler "NoSuchKey".

Example

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.NoSuchKey</Faultcode>
    <Faultstring>The specified key does not exist.</Faultstring>
    <Detail>
      <Key>Fred</Key>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

Weitere Informationen zu Amazon-S3-Fehlern finden Sie unter ErrorCodeList.

Bewährte Methoden für Amazon-S3-Fehler

Wenn Sie eine Anwendung für Amazon S3 entwerfen, sollten Sie darauf achten, dass Amazon-S3-Fehler auf geeignete Weise verarbeitet werden. Dieser Abschnitt beschreibt Aspekte, die Sie beim Entwurf Ihrer Anwendung berücksichtigen sollten.

Wiederholung bei InternalErrors

Interne Fehler sind Fehler, die innerhalb der Amazon-S3-Umgebung auftreten.

Anfragen, die eine InternalError-Antwort erhalten, wurden möglicherweise nicht verarbeitet. Gibt eine PUT-Anfrage beispielsweise einen InternalError zurück, ruft ein nachfolgendes GET möglicherweise den alten Wert oder den aktuellen Wert ab.

Wenn Amazon S3 eine InternalError-Antwort zurückgibt, wiederholen Sie die Anforderung.

Optimieren der Anwendung für wiederholte SlowDown-Fehler

Wie jedes verteilte System besitzt S3 Schutzmechanismen, die einen beabsichtigten oder nicht beabsichtigten übermäßigen Ressourcenverbrauch erkennen und entsprechend reagieren.

SlowDown-Fehler können auftreten, wenn eine hohe Anfragerate einen dieser Mechanismen auslöst. Eine Reduzierung Ihrer Anfragerate verringert oder eliminiert Fehler dieses Typs. Im Allgemeinen werden die meisten Benutzer diese Fehler nicht regelmäßig sehen. Wenn Sie jedoch weitere Informationen wünschen oder viele oder unerwartete SlowDown-Fehler sehen, posten Sie bitte in unserem Amazon-S3-Entwicklerforum oder melden Sie sich für AWS Support an https://aws.amazon.com/premiumsupport/.

Isolieren von Fehlern

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Amazon S3 unterstützt verschiedene Fehlercodes, die sowohl von der SOAP, als auch von der REST-API verwendet werden. Die SOAP API gibt Amazon-S3-Standardfehlercodes zurück. Die REST-API ist darauf ausgelegt, wie ein HTTP-Standardserver auszusehen und mit vorhandenen HTTP-Clients zu interagieren (z. B. Browsern, HTTP-Clientbibliotheken, Proxies, Caches usw.). Um sicherzustellen, dass die HTTP-Clients Fehler ordnungsgemäß verarbeiten, bilden wir jeden Amazon-S3-Fehler auf einen HTTP-Statuscode ab.

HTTP-Statuscodes sind weniger ausdrucksstark als Amazon-S3-Fehlercodes und enthalten weniger Informationen über den Fehler. Beispielsweise werden die Amazon-S3-Fehler `NoSuchKey` und `NoSuchBucket` auf den Statuscode `HTTP 404 Not Found` abgebildet.

Obwohl HTTP-Statuscodes weniger Informationen über den Fehler enthalten, können Clients, die HTTP verstehen, aber nicht die Amazon-S3-API, den Fehler normalerweise korrekt verarbeiten.

Wenn Sie also Fehler verarbeiten oder Amazon-S3-Fehler an Endbenutzer melden, verwenden Sie den Amazon-S3-Fehlercode statt des HTTP-Statuscodes, weil dieser die meisten Informationen über den Fehler enthält. Beim Debuggen Ihrer Anwendung sollten Sie außerdem das lesbare Element <Details> der XML-Fehlerantwort heranziehen.

Referenz für Entwickler

Dieser Anhang umfasst folgende Abschnitte.

Themen

Anhang A: Verwenden der SOAP-API

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Dieser Abschnitt enthält spezifische Informationen für die Amazon S3 SOAP-API.

> **Note**
>
> SOAP-Anfragen, und zwar sowohl authentifizierte als auch anonyme, müssen mit SSL an Amazon S3 gesendet werden. Amazon S3 gibt einen Fehler zurück, wenn Sie eine SOAP-Anfrage über HTTP senden.

Themen

Allgemeine SOAP-API-Elemente

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Sie können mit Amazon S3 unter Verwendung von SOAP 1.1 über HTTP interagieren. Die Amazon S3 WSDL, die die Amazon-S3-API maschinenlesbar beschreibt, ist unter folgender Adresse verfügbar: https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl. Das Amazon-S3-Schema ist unter https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.xsd verfügbar.

Die meisten Benutzer interagieren mit Amazon S3 über ein SOAP-Toolkit, das auf ihre Sprache und Entwicklungsumgebung zugeschnittenen ist. Verschiedene Toolkits machen die Amazon-S3-API auf unterschiedliche Weise verfügbar. Bitte konsultieren Sie Ihre spezifische Toolkit-Dokumentation für seine Verwendung. Dieser Abschnitt veranschaulicht die Amazon-S3-SOAP-Operationen in einer Toolkit-unabhängigen Weise durch Darstellen der XML-Anfragen und Antworten, wie sie „online“ erscheinen.

Allgemeine Elemente

Sie können die folgenden autorisierungsbezogenen Elemente in jede SOAP-Anfrage aufnehmen:

- `AWSAccessKeyId:` Die AWS-Zugriffsschlüssel-ID des Anforderers
- `Timestamp:` Die aktuelle Systemzeit
- `Signature:` Die Signatur für die Anforderung

Authentifizieren von SOAP-Anforderungen

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Jede nicht anonyme Anfrage muss Authentifizierungsinformationen enthalten, um die Identität des Prinzipals einzurichten, der die SOAP-Anfrage stellt. In SOAP wird die Authentifizierungsinformation in den folgenden Elemente der SOAP-Anfrage abgelegt:

- Ihre AWS-Zugriffsschlüssel-ID.

> **Note**
>
> Bei authentifizierten SOAP-Anfragen werden keine temporären Sicherheitsanmeldeinformationen unterstützt. Weitere Informationen den verschiedenen Anmeldeinformationen finden Sie unter Senden von Anforderungen.

- `Timestamp:` Dabei muss es sich um ein dateTime (siehe http://www.w3.org/TR/xmlschema-2/#dateTime) in der Zeitzone Coordinated Universal Time (Greenwich Mean Time) handeln, beispielsweise `2009-01-01T12:00:00.000Z`. Die Autorisierung schlägt fehl, wenn dieser Zeitstempel um mehr als 15 Minuten von der Uhr auf Amazon-S3-Servern abweicht.
- `Signature:` Der RFC 2104 HMAC-SHA1-Digest (siehe http://www.ietf.org/rfc/rfc2104.txt) der Verkettung von "AmazonS3" + OPERATION + Zeitstempel, wobei Ihr geheimer AWS-Zugriffsschlüssel als Schlüssel verwendet wird. In der folgenden CreateBucket-Beispielanfrage würde das Signatur-Element den HMAC-SHA1-Digest mit dem Wert "AmazonS3CreateBucket2009-01-01T12:00:00.000Z" enthalten:

In der folgenden CreateBucket-Beispielanfrage würde das Signatur-Element den HMAC-SHA1-Digest mit dem Wert "AmazonS3CreateBucket2009-01-01T12:00:00.000Z" enthalten:

Example

```
<CreateBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Acl>private</Acl>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2009-01-01T12:00:00.000Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</CreateBucket>
```

> **Note**
>
> SOAP-Anfragen, und zwar sowohl authentifizierte als auch anonyme, müssen mit SSL an Amazon S3 gesendet werden. Amazon S3 gibt einen Fehler zurück, wenn Sie eine SOAP-Anfrage über HTTP senden.

> **Important**
>
> Aufgrund von unterschiedlichen Interpretationen dahingehend, wie eine zusätzliche Zeitgenauigkeit wegfallen kann, sollten .NET-Benutzer darauf achten, Amazon S3 keine übermäßig spezifischen Zeitstempel zu senden. Dies kann bewerkstelligt werden, indem manuell `DateTime`-Objekte mit einer Millisekunde Genauigkeit erstellt werden.

Festlegen der Zugriffsrichtlinie mit SOAP

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Die Zugriffskontrolle kann zu dem Zeitpunkt eingerichtet werden, zu dem ein Bucket oder ein Objekt geschrieben werden, einschließlich des „AccessControlList“-Elements, indem `CreateBucket`, `PutObjectInline` oder `PutObject` aufgerufen werden. Das AccessControlList-Element ist in beschriebe Identity and Access Management in Amazon S3. Wenn für diese Operationen keine Zugriffskontrollliste angegeben ist, wird die Ressource mit einer Standard-Zugriffsrichtlinie erstellt, die dem Auftraggeber FULL_CONTROL-Zugriff erteilt (das erfolgt auch dann, wenn es sich um eine PutObjectInline- oder PutObject-Anfrage für ein bereits vorhandenes Objekt handelt).

Nachfolgend sehen Sie eine Anfrage, die Daten in ein Objekt schreibt, das Objekt lesbar für anonyme Prinzipale macht, und dem angegebenen Benutzer FULL_CONTROL-Rechte für den Bucket erteilt (die meisten Entwickler wollen sich selbst FULL_CONTROL-Zugriff auf ihren eigenen Bucket geben).

Example

Die nachfolgende Anfrage schreibt Daten in ein Objekt und macht das Objekt lesbar für anonyme Prinzipale.

`Sample Request`

```
<PutObjectInline xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
```

```
  <Key>Nelson</Key>
  <Metadata>
    <Name>Content-Type</Name>
    <Value>text/plain</Value>
  </Metadata>
  <Data>aGEtaGE=</Data>
  <ContentLength>5</ContentLength>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers<URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2009-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</PutObjectInline>
```

Sample Response

```
<PutObjectInlineResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <PutObjectInlineResponse>
    <ETag>&quot828ef3fdfa96f00ad9f27c383fc9ac7f&quot</ETag>
    <LastModified>2009-01-01T12:00:00.000Z</LastModified>
  </PutObjectInlineResponse>
</PutObjectInlineResponse>
```

Die Zugriffskontrollrichtlinie kann für einen vorhandenen Bucket oder ein vorhandenes Objekt mit den Methoden `GetBucketAccessControlPolicy`, `GetObjectAccessControlPolicy`, `SetBucketAccessControlPolicy` und `SetObjectAccessControlPolicy` gelesen oder eingerichtet werden. Weitere Informationen finden Sie in der detaillierten Erklärung dieser Methoden.

Anhang B: Authentifizieren von Anforderungen (AWS Signature Version 2)

Important

In diesem Abschnitt wird das Authentifizieren von Anforderungen unter Verwendung von AWS Signature Version 2 beschrieben. Signature Version 2 wird deaktiviert (veraltet), Amazon S3 akzeptiert nur API-Anforderungen, die mit Signature Version 4 signiert wurden. Weitere Informationen finden Sie unter AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet)

Signature Version 4 wird in allen AWS-Regionen unterstützt und ist die einzige Version, die in neuen Regionen unterstützt wird. Weitere Informationen finden Sie unter Authenticating Requests (Authentifizierung von Anforderungn) (AWS Signature Version 4) in der API-Referenz für Amazon Simple Storage Service.

Amazon S3 ermöglicht das Ermitteln der zum Signieren einer Anforderung verwendeten API Signature Version. Um Beeinträchtigungen der betrieblichen Prozesse zu vermeiden, muss festgestellt werden, ob Workflows mit Signature Version 2 signieren. Diese Workflows müssen dann auf die Verwendung von Signature Version 4 umgestellt werden.

- Wenn Sie CloudTrail-Ereignisprotokolle verwenden (empfohlen), finden Sie unter Identifizieren von Anforderungen von Amazon S3 Signature Version 2 mithilfe von CloudTrail Informationen zum Abfragen und Identifizieren solcher Anforderungen.
- Wenn Sie die Amazon S3 Server Access-Protokolle verwenden, vgl. Identifizieren von Signature-Version-2-Anforderungen mittels Amazon-S3-Zugriffsprotokollen

Themen

Authentifizieren von Anforderungen mit der REST-API

Beim Zugriff auf Amazon S3 mit REST müssen Sie in Ihrer Anfrage die folgenden Elemente angeben, damit diese authentifiziert werden kann.:

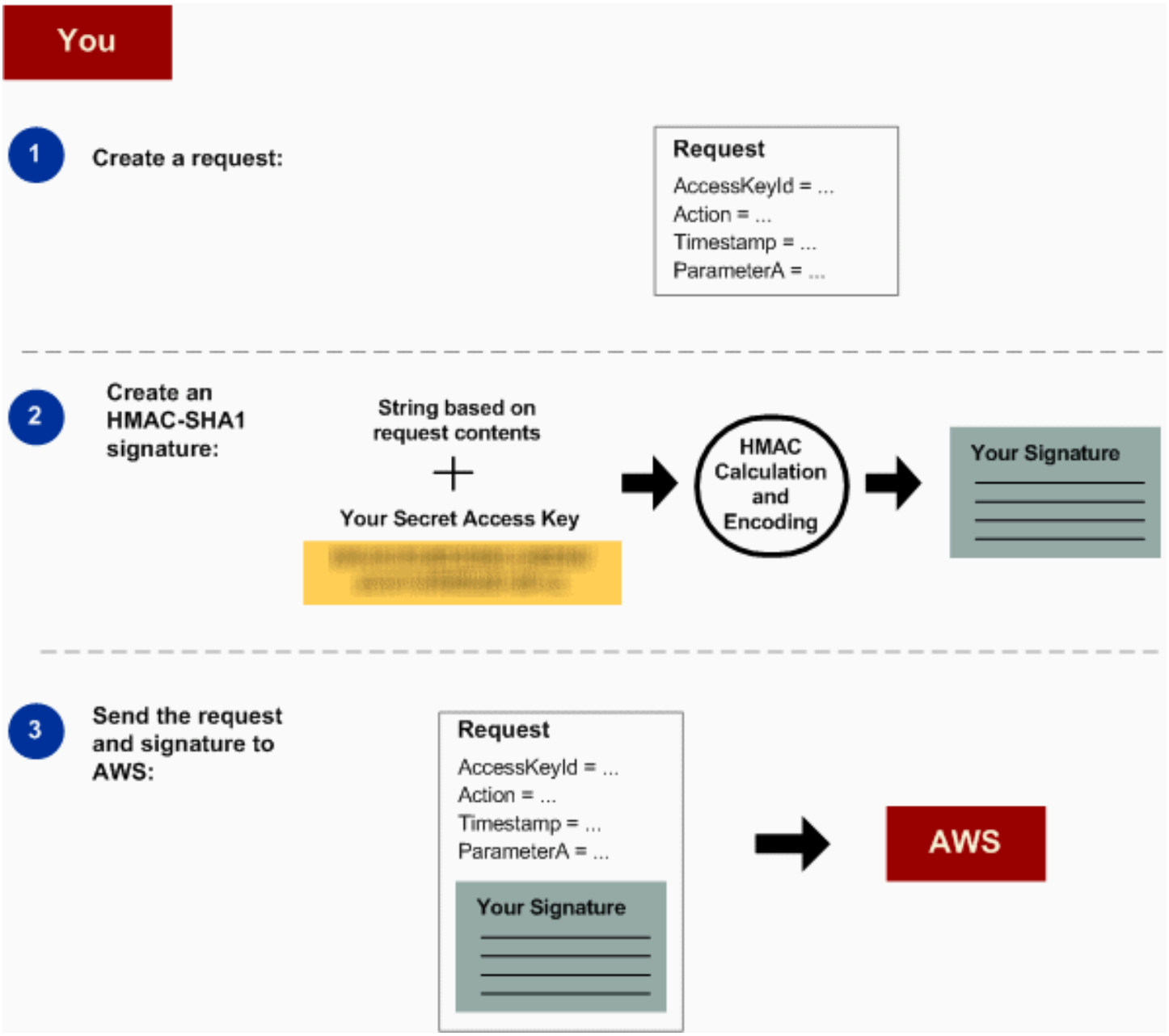
Anfordern von Elementen

- AWS-Zugriffsschlüssel-ID – Jede Anfrage muss die Zugriffsschlüssel-ID der Identität enthalten, mit der Sie Ihre Anfrage senden.
- Signatur – Jede Anforderung muss eine gültige Anforderungsignatur enthalten, oder die Anforderung wird abgelehnt.

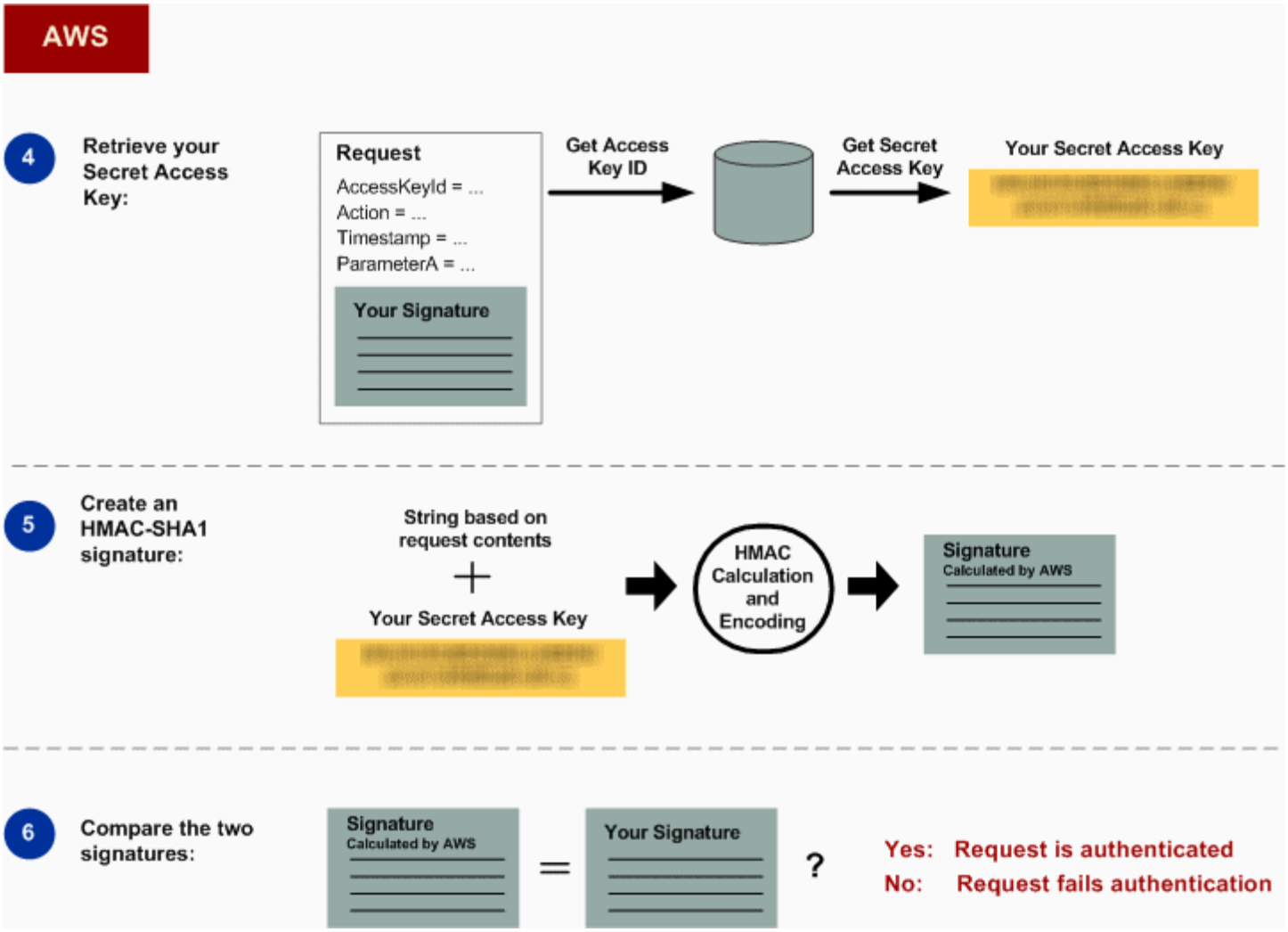
  Eine Anforderungsignatur wird unter Verwendung Ihres geheimen Zugriffsschlüssels berechnet, das ist ein gemeinsames Geheimnis, das nur Sie und AWS kennen.
- Zeitstempel – Jede Anfrage muss Datum und Uhrzeit des Zeitpunktes enthalten, zu dem die Anfrage erstellt wurde, dargestellt als Zeichenfolge in UTC.
- Datum – Jede Anfrage muss den Zeitstempel der Anfrage enthalten.

  Abhängig von der verwendeten API-Aktion können Sie ein Ablaufdatum und eine Ablaufzeit für die Anfrage angeben, statt dem Zeitstempel oder zusätzlich zu diesem. Um festzustellen, was notwendig ist, lesen Sie im Authentifizierungsthema für die jeweiligen Aktion nach.

Nachfolgend finden Sie die allgemeinen Schritte für die Authentifizierung von Anfragen für Amazon S3. Es wird vorausgesetzt, dass Sie über die erforderlichen Sicherheitsanmeldeinformationen, die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel verfügen.



1	Erstellen einer Anfrage an AWS.
2	Berechnung der Signatur unter Verwendung Ihres geheimen Zugriffsschlüssels.
3	Senden Sie die Anfrage an Amazon S3. Nehmen Sie Ihre Zugriffsschlüssel-ID und die Signatur in Ihre Anforderung auf. Amazon S3 führt die nächsten drei Schritte aus.



4	Amazon S3 verwendet die Zugriffsschlüssel-ID, um Ihren geheimen Zugriffschlüssel abzurufen.
5	Amazon S3 berechnet eine Signatur aus den Anforderungdaten und dem geheimen Zugriffsschlüssel mit demselben Algorithmus, mit dem die in der Anforderung gesendete Signatur berechnet wurde.
6	Wenn die von Amazon S3 generierte Signatur mit der in der Anforderung gesendete n Signatur übereinstimmt, wird die Anforderung als authentisch eingestuft. Falls der Vergleich fehlschlägt, wird die Anfrage verworfen, und Amazon S3 gibt eine Fehlerant wort zurück.

Detaillierte Authentifizierungsdaten

Weitere Informationen zur REST-Authentifizierung finden Sie unter Signieren und Authentifizieren von REST-Anforderungen.

Signieren und Authentifizieren von REST-Anforderungen

Themen

> **Note**
>
> Dieses Thema erklärt Authentifizierungsanfragen unter Verwendung von Signature Version 2. Amazon S3 unterstützt jetzt die neueste Signature Version 4. Diese neueste Signatur-Version wird in allen Regionen unterstützt. Alle neuen Regionen unterstützen nach dem 30. Januar 2014 nur Signature Version 4. Weitere Informationen finden Sie unter Authenticating Requests (Authentifizierung von Anforderungn) (AWS Signature Version 4) in der API-Referenz für Amazon Simple Storage Service.

Die Authentifizierung ist der Prozess, Ihre Identität gegenüber dem System nachzuweisen. Die Identität ist ein wichtiger Faktor in den Zugriffssteuerungsentscheidungen von Amazon S3. Anforderungen werden zum Teil basierend auf der Identität des Auftraggebers zugelassen oder abgewiesen. Das Recht, Buckets zu erstellen, ist beispielsweise für registrierte Entwickler reserviert (standardmäßig), und das Recht, Objekte in einem Bucket zu erstellen, ist für den Eigentümer des betreffenden Buckets reserviert. Als Entwickler machen Sie Anforderungen, für die diese

Berechtigungen gelten müssen, deshalb müssen Sie Ihre Identität gegenüber dem System belegen, indem Sie Ihre Anforderungen authentifizieren. In diesem Abschnitt erfahren Sie mehr darüber.

> **Note**
>
> Der Inhalt dieses Abschnitts gilt nicht für HTTP POST. Weitere Informationen finden Sie unter Browserbasierte Uploads mit POST (AWS Signature Version 2).

Die Amazon-S3-REST-API verwendet ein allgemeines HTTP-Schema basierend auf einem verschlüsselten HMAC (Hash Message Authentication Code) für die Authentifizierung. Um eine Anforderung zu authentifizieren, verknüpfen Sie zuerst ausgewählte Elemente der Anforderung, um eine Zeichenfolge zu erstellen. Anschließend verwenden Sie Ihren geheimen AWS-Zugriffsschlüssel, um den HMAC für diese Zeichenfolge zu berechnen. Informell bezeichnen wir diesen Prozess als „Signieren der Anforderung“. Wir bezeichnen die Ausgabe des HMAC-Algorithmus als die Signatur, weil sie die Sicherheitseigenschaften einer realen Signatur simuliert. Schließlich fügen Sie diese Signatur als Parameter der Anforderung hinzu. Dazu verwenden Sie die in diesem Abschnitt beschriebene Syntax.

Wenn das System eine authentifizierte Anforderung erhält, lädt es den geheimen AWS-Zugriffsschlüssel, den Sie behaupten zu haben, und verwendet ihn genau so, um aus der empfangenen Nachricht eine Signatur zu berechnen. Anschließend vergleicht es die berechnete Signatur mit der Signatur, die der Anforderer vorgezeigt hat. Stimmen die beiden Signaturen überein, schließt das System daraus, dass der Anforderer Zugriff auf den geheimen AWS-Zugriffsschlüssel hat, und damit mit der Genehmigung des Prinzipals handelt, dem der Schlüssel ausgestellt wurde. Stimmen die beiden Signaturen nicht überein, wird die Anforderung verworfen und das System antwortet mit einer Fehlermeldung.

Example Authentifizierte Amazon S3 REST-Anfrage

```
GET /photos/puppy.jpg HTTP/1.1
Host: awsexamplebucket1.us-west-1.s3.amazonaws.com
Date: Tue, 27 Mar 2007 19:36:42 +0000

Authorization: AWS AKIAIOSFODNN7EXAMPLE:
qgk2+6Sv9/oM7G3qLEjTH1a1l1g=
```

Verwenden von temporären Sicherheitsanmeldeinformationen

Wenn Sie Ihre Anfrage unter Verwendung temporärer Sicherheitsanmeldeinformationen signieren (siehe Senden von Anforderungen), müssen Sie das entsprechende Sicherheitstoken in Ihre Anfrage aufnehmen, indem Sie den `x-amz-security-token`-Header hinzufügen.

Wenn Sie unter Verwendung der AWS Security Token Service API temporäre Sicherheitsanmeldeinformationen erhalten haben, beinhaltet die Antwort temporäre Sicherheitsanmeldeinformationen und ein Sitzungstoken. Sie geben den Wert des Sitzungstokens im `x-amz-security-token`-Header an, wenn Sie Anfragen an Amazon S3 senden. Informationen zur von IAM bereitgestellten AWS Security Token Service-API finden Sie unter Aktions (Aktionen) im AWS Security Token Service-API-Referenzhandbuch.

Der Authentifizierungsheader

Die Amazon-S3-REST-API verwendet den standardmäßigen HTTP-`Authorization`-Header, um Authentifizierungs-Informationen weiterzugeben. (Der Name des Standardheaders ist unglücklich gewählt, weil er eine Authentifizierung weitergibt, keine Autorisierung.) Unter dem Amazon-S3-Authentifizierungsschema hat der Autorisierungsheader die folgende Form:

```
Authorization: AWS AWSAccessKeyId:Signature
```

Entwicklern wird eine AWS-Zugriffsschlüssel-ID und ein geheimer AWS-Zugriffsschlüssel ausgestellt, wenn sie sich anmelden. Für die Anforderungs-Authentifizierung identifiziert das `AWSAccessKeyId`-Element die Zugriffsschlüssel-ID, die für die Berechnung der Signatur verwendet wurde, und indirekt für den Entwickler steht, der die Anforderung gestellt hat.

Das `Signature`-Element ist das RFC 2104 HMAC-SHA1 ausgewählter Elemente aus der Anforderung, der `Signature`-Teil des Authorization-Headers unterscheidet sich deshalb zwischen verschiedenen Anforderungen. Wenn die vom System berechnete Anforderungssignatur mit der in der Anforderung enthaltenen `Signature` übereinstimmt, hat der Auftraggeber belegt, dass er den geheimen AWS-Zugriffsschlüssel besitzt. die Anforderung wird unter der Identität verarbeitet, und mit der Genehmigung des Entwicklers, dem der Schlüssel ausgestellt wurde.

Die folgende Pseudogrammatik verdeutlicht den Aufbau des `Authorization`-Anforderungs-Headers. (In dem Beispiel steht \n für den Unicode-Code `U+000A`, häufig auch als Newline bezeichnet.)

```
Authorization = "AWS" + " " + AWSAccessKeyId + ":" + Signature;
```

```
Signature = Base64( HMAC-SHA1( UTF-8-Encoding-Of(YourSecretAccessKey), UTF-8-Encoding-
Of( StringToSign ) ) );

StringToSign = HTTP-Verb + "\n" +
 Content-MD5 + "\n" +
 Content-Type + "\n" +
 Date + "\n" +
 CanonicalizedAmzHeaders +
 CanonicalizedResource;

CanonicalizedResource = [ "/" + Bucket ] +
 <HTTP-Request-URI, from the protocol name up to the query string> +
 [ subresource, if present. For example "?acl", "?location", or "?logging"];

CanonicalizedAmzHeaders = <described below>
```

HMAC-SHA1 ist ein von RFC 2104 - Keyed-Hashing for Message Authentication definierter Algorithmus. Der Algorithmus nimmt zwei Byte-Zeichenfolgen als Eingabe entgegen, einen Schlüssel und eine Nachricht. Für die Amazon-S3-Anfrageauthentifizierung verwenden Sie Ihren geheimen AWS-Zugriffsschlüssel (`YourSecretAccessKey`) als Schlüssel und die UTF-8-Codierung des `StringToSign` als die Meldung. Die Ausgabe von HMAC-SHA1 ist ebenfalls eine Byte-Zeichenfolge, der sogenannte Digest. Der `Signature`-Anforderungsparameter wird durch eine Base64-Codierung dieses Digest erstellt.

Anfordern einer Kanonisierung für die Signatur

Wenn das System eine authentifizierte Anforderung erhält, vergleicht es die berechnete Anforderungssignatur mit der in der Anforderung im breitgestellten Signatur, wie bereits beschriebe `StringToSign`. Aus diesem Grund müssen Sie die Signatur nach derselben Methode berechnen, die auch Amazon S3 verwendet. Wir bezeichnen diesen Prozess, bei dem eine Anforderung in die für die Signatur vereinbarte Form gebracht wird, als Kanonisierung.

Konstruieren des - CanonicalizedResource Elements

`CanonicalizedResource` stellt die für die Anfrage relevante Amazon-S3-Ressource dar. Für eine REST-Anforderung erstellen Sie sie wie folgt:

Starten des Prozesses

1	Beginnen Sie mit einer leeren Zeichenfolge ("").

2	Wenn die Anforderung unter Verwendung des HTTP Host-Headers (virtueller gehosteter Stils) einen Bucket angibt, fügen Sie den Bucket-Namen mit vorgestelltem "/" an (z. B. „/bucketname“). Für Anforderungen im Pfad-Stil und Anforderungen, die an keinen spezifischen Bucket gerichtet sind, machen Sie nichts. Weitere Informationen zum Wiederholen von Anforderungen finden Sie unter Virtuelles Hosting bei Buckets. Für die Virtual Hosted-Anforderung „https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg“ lautet die `CanonicalizedResource` „/awsexamplebucket1“. Für die Path-Anforderung „https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg“ lautet `CanonicalizedResource` „“.
3	Fügen Sie den Pfad-Abschnitt der nicht decodierten HTTP Anforderungs-URI bis zur (aber nicht inklusive der) Abfragezeichenfolge ein. Für die Virtual Hosted-Anforderung „https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg“ lautet die `CanonicalizedResource` „/awsexamplebucket1/photos/puppy.jpg“. Für die Path-Anforderung „https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg“ lautet die `CanonicalizedResource` „/awsexamplebucket1/photos/puppy.jpg“. An dieser Stelle ist der `CanonicalizedResource` für Anforderungen mit virtuell gehostetem Stil und Pfad-Stil gleich. Im Fall einer Anforderung, die sich nicht an einen Bucket richtet, wie GET Service, fügen Sie „/“ an.

4	Wenn sich die Anforderung an eine Subressource richtet, wie beispielsweise `?versioning` , `?location` , `?acl`, `?lifecycle` , oder `?versionid` , fügen Sie die Subressource an, gegebenenfalls ihren Wert und das Fragezeichen. Beachten Sie, dass mehrere Subressourcen alphabetisch nach dem Namen der Subressource sortiert und durch '&' voneinander getrennt werden müssen, z. B. ?acl&versionId=*value*. Die Subressourcen, die beim Erstellen des CanonicalizedResource Elements enthalten sein müssen, sind acl, lifecycle, location, logging, notification, partNumber, policy, requestPayment ,uploadId, uploads, versionId, versioning, versions und website. Wenn die Anfrage Abfragezeichenfolgen-Parameter angibt, die die Antwort-Header-Werte überschreiben (siehe Get Object), fügen Sie die Abfragezeichenfolgen-Parameter und ihre Werte an. Bei einer Signatur codieren Sie diese Werte nicht. Bei einer Anforderung dagegen müssen Sie diese Parameterwerte codieren. Die Abfrageparameter in einer GET-Anforderung sind unter anderem `response-content-type` , `response-content-language` , `response-expires` , `response-cache-control` , `response-content-disposition` und `response-content-encoding` . Der `delete` Abfragezeichenfolgeparameter muss enthalten sein, wenn Sie die CanonicalizedResource für eine Anforderung zum Löschen mehrerer Objekte erstellen.

Elemente der CanonicalizedResource , die aus der HTTP Request-URI stammen, sollten so signiert werden, wie sie in der HTTP-Anforderung erscheinen, einschließlich URL-Kodierungsmetazeichen.

Möglicherweise unterschiedet sich die `CanonicalizedResource` von der HTTP Anforderungs-URI. Wenn Ihre Anforderung den HTTP-Header `Host` verwendet, um einen Bucket anzugeben, erscheint der Bucket nicht in der HTTP Anforderungs-URI. Die `CanonicalizedResource` beinhaltet den Bucket jedoch weiterhin. Abfrage-Zeichenfolgenparameter können auch in der Anforderungs-URI erscheinen, sind aber nicht in der enthalte `CanonicalizedResource`. Weitere Informationen finden Sie unter Virtuelles Hosting bei Buckets.

Konstruieren des - CanonicalizedAmzHeaders Elements

Um den CanonicalizedAmzHeaders Teil von zu erstellen`StringToSign`, wählen Sie alle HTTP-Anforderungsheader aus, die mit „x-amz-“ beginnen (unter Verwendung eines Vergleichs ohne Berücksichtigung der Groß- und Kleinschreibung), und gehen Sie wie folgt vor.

CanonicalizedAmzHeaders Prozess

1	Wandeln Sie jeden HTTP-Headernamen in Kleinbuchstaben um. Beispielsweise wird 'X-Amz-Date ' zu 'x-amz-date '.
2	Sortieren Sie die Header alphabetisch nach dem Headernamen.
3	Kombinieren Sie Header-Felder mit demselben Namen zu einem „header-name:comma -separated-value-list“-Paar, wie in RFC 2616, Abschnitt 4.2, ohne Leerzeichen zwischen Werten vorgeschrieben. Die beiden Metadaten-Header 'x-amz-meta-username: fred' und 'x-amz-meta-username: barney ' würden beispielsweise zu einem einzigen Header 'x-amz-meta-username: fred,barney ' zusammengefasst.
4	„Falten“ Sie lange Header „auf“, die sich über mehrere Zeilen erstrecken (wie durch RFC 2616 Abschnitt 4.2 erlaubt), indem Sie die „faltenden“ Leerzeichen (einschließlich Neue-Zeil e-Zeichen) durch ein einzelnes Leerzeichen ersetzen.
5	Löschen Sie alle Leerzeichen um den Doppelpunkt in der Kopfzeile. Beispielsweise würde der Header 'x-amz-meta-username: fred,barney ' zu 'x-amz-meta-usernam e:fred,barney '
6	Schließlich fügen Sie jedem kanonisierten Header in der Ergebnisliste ein Neue-Zeile-Zeichen (U+000A) hinzu. Konstruieren Sie das - CanonicalizedResource Element, indem Sie alle Header in dieser Liste zu einer einzigen Zeichenfolge verketten.

Positionale und benannte HTTP-Header- StringToSign Elemente

Die ersten Header-Elemente von StringToSign (Content-Type, Date und Content-MD5) sind von ihrer Art her positionsbezogen. StringToSign fügt nicht die Namen dieser Header, sondern nur ihre Werte aus der Anforderung ein. Im Gegensatz dazu sind die 'x-amz-'-Elemente benannt. Sowohl die Header-Namen als auch die Header-Werte erscheinen in StringToSign.

Wenn ein positionaler Header, der in der Definition von StringToSign vorgegeben ist, in Ihrer Anforderung nicht vorhanden ist (z. B. sind Content-Type oder Content-MD5 optionale für PUT-Anforderungen und sinnlos für GET-Anforderungen), setzen Sie in dieser Position die leere Zeichenfolge ("") ein.

Zeitstempel-Anforderung

Ein gültiger Zeitstempel (unter Verwendung des HTTP-Headers `Date` oder einer `x-amz-date`-Alternative) ist zwingend erforderlich für authentifizierte Anforderungen. Darüber hinaus muss der Client-Zeitstempel in einer authentifizierten Anfrage innerhalb eines Zeitraums von 15 Minuten zur Amazon-S3-Systemzeit liegen, wenn die Anfrage empfangen wird. Wenn dies nicht der Fall ist, schlägt die Anforderung mit `RequestTimeTooSkewed`-Fehlercode fehl. Diese Einschränkungen sollen verhindern, dass abgefangene Anforderungen böswillig wiederholt werden. Für einen strengeren Schutz gegen ein Abhören transportieren Sie authentifizierte Anforderung mit HTTPS.

> **Note**
>
> Die Auswertungsbeschränkung im Hinblick auf das Anforderungsdatum gilt nur für authentifizierte Anforderungen, die keine Abfrage-String-Authentifizierung verwenden. Weitere Informationen finden Sie unter Alternative zur Authentifizierung der Abfragezeichenfolge.

Einige HTTP-Client-Bibliotheken unterstützen die Möglichkeit nicht, den `Date`-Header für eine Anforderung einzurichten. Wenn Sie Probleme damit haben, den Wert des 'Date'-Headers in kanonisierte Header aufzunehmen, können Sie den Zeitstempel für die Anforderung stattdessen auch mit einem '`x-amz-date`'-Header einrichten. Der Wert des `x-amz-date`-Headers muss eines der RFC 2616-Formate haben (http://www.ietf.org/rfc/rfc2616.txt). Wenn in einer Anforderung ein `x-amz-date`-Header vorhanden ist, ignoriert das System bei der Berechnung der Anforderungssignatur jeden `Date`-Header. Wenn Sie also den `x-amz-date`-Header aufnehmen, verwenden Sie die leere Zeichenfolge für `Date`, wenn Sie den `StringToSign` erstellen. Ein Beispiel finden Sie im nächsten Abschnitt.

Authentifizierungsbeispiele

Die Beispiele in diesem Abschnitt verwenden die (nicht funktionierenden) Anmeldeinformationen aus der folgenden Tabelle.

Parameter	Wert
AWSAccessKeyId	`AKIAIOSFODNN7EXAMPLE`
AWSSecret AccessKey	`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`

In den `StringToSigns` des Beispiels ist die Formatierung nicht relevant, und `\n` steht für den Unicode-Punkt `U+000A`, allgemein als Neue-Zeile-Zeichen bezeichnet. Außerdem verwenden die Beispiele "+0000", um die Zeitzone anzugeben. Sie können die Zeitzone stattdessen mit "GMT" angeben, aber in den Beispielen werden andere Signaturen verwendet.

Object GET

In diesem Beispiel wird ein Objekt aus dem Bucket „awsexamplebucket1“ abgerufen.

Anforderung	StringToSign
`GET /photos/puppy.jpg HTTP/1.1` `Host: awsexamplebucket1.us-west-1.s3.amazonaws.com` `Date: Tue, 27 Mar 2007 19:36:42 +0000` `Authorization: AWS AKIAIOSFODNN7EXAMPLE:qgk2+6Sv9/oM7G3qLEjTH1a1l1g=`	`GET\n` `\n` `\n` `Tue, 27 Mar 2007 19:36:42 +0000\n` `/awsexamplebucket1/photos/puppy.jpg`

Beachten Sie, dass die den Bucket-Namen CanonicalizedResource enthält, die HTTP Request-URI jedoch nicht. (Der Bucket wird vom Host-Header spezifiziert.)

> **Note**
>
> Das folgende Python-Skript berechnet die vorhergehende Signatur unter Verwendung der bereitgestellten Parameter. Sie können dieses Skript verwenden, um Ihre eigenen Signaturen zu erstellen und die Schlüssel und StringToSign nach Bedarf zu ersetzen.

```
import base64
import hmac
from hashlib import sha1

access_key = 'AKIAIOSFODNN7EXAMPLE'.encode("UTF-8")
secret_key = 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY'.encode("UTF-8")

string_to_sign = 'GET\n\n\nTue, 27 Mar 2007 19:36:42 +0000\n/awsexamplebucket1/
photos/puppy.jpg'.encode("UTF-8")
signature = base64.b64encode(
```

```
                                hmac.new(
                                        secret_key, string_to_sign, sha1
                                        ).digest()
                                ).strip()


print(f"AWS {access_key.decode()}:{signature.decode()}")
```

Object PUT

In diesem Beispiel wird ein Objekt in den Bucket „awsexamplebucket1“ eingefügt.

Anforderung	StringToSign
`PUT /photos/puppy.jpg HTTP/1.1` `Content-Type: image/jpeg` `Content-Length: 94328` `Host: awsexamplebucket1.s3.us-wes` `t-1.amazonaws.com` `Date: Tue, 27 Mar 2007 21:15:45 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMP`* *`LE:`* *`iqRzw+ileNPu1fhspnRs8nOjjIA=`*	`PUT\n` `\n` `image/jpeg\n` `Tue, 27 Mar 2007 21:15:45 +0000\n` `/awsexamplebucket1/photos/puppy.jpg`

Notieren Sie sich den Content-Type-Header in der Anforderung und in der StringToSign. Beachten Sie auch, dass Content-MD5 in der leer gelassen wird StringToSign, da es in der Anforderung nicht vorhanden ist.

Auflisten

In diesem Beispiel wird der Inhalt des Buckets „awsexamplebucket1“ aufgelistet.

Anforderung	StringToSign
`GET /?prefix=photos&max-keys=50&marker=puppy` `HTTP/1.1`	`GET\n` `\n`

Anforderung	StringToSign
`User-Agent: Mozilla/5.0` `Host: awsexamplebucket1.s3.us-west-1.amazo` `naws.com` `Date: Tue, 27 Mar 2007 19:42:41 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE:`* *`m0WP8eCtspQl5Ahe6L1SozdX9YA=`*	`\n` `Tue, 27 Mar 2007 19:42:41` `+0000\n` `/awsexamplebucket1/`

Notieren Sie sich den abschließenden Schrägstrich in der CanonicalizedResource und das Fehlen von Abfragezeichenfolgeparametern.

Fetch

In diesem Beispiel wird die Zugriffskontrollrichtlinien-Subressource für den Bucket „awsexamplebucket1“ abgerufen.

Anforderung	StringToSign
`GET /?acl HTTP/1.1` `Host: awsexamplebucket1.s3.us-west-1.amazo` `naws.com` `Date: Tue, 27 Mar 2007 19:44:46 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE:`* *`82ZHiFIjc+WbcwFKGUVEQspPn+0=`*	`GET\n` `\n` `\n` `Tue, 27 Mar 2007 19:44:46` `+0000\n` `/awsexamplebucket1/?acl`

Beachten Sie, wie der Subressourcen-Abfragezeichenfolgeparameter in der enthalten ist CanonicalizedResource.

Löschen

In diesem Beispiel wird ein Objekt über die path-style- und Date-Alternative aus dem Bucket „awsexamplebucket1“ entfernt.

Anforderung	StringToSign
`DELETE /awsexamplebucket1/photos/puppy.jpg HTTP/1.1` `User-Agent: dotnet` `Host: s3.us-west-1.amazonaws.com` `Date: Tue, 27 Mar 2007 21:20:27 +0000` `x-amz-date: Tue, 27 Mar 2007 21:20:26 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE:XbyTlbQdu9Xw5o8P4iMwPktxQd8=`*	`DELETE\n` `\n` `\n` `Tue, 27 Mar 2007 21:20:26 +0000\n` `/awsexamplebucket1/photos/puppy.jpg`

Beachten Sie, wie wir die alternative Methode „x-amz-date“ zur Angabe des Datums verwendet haben (weil unsere Client-Bibliothek verhindert hat, dass wir das Datum festlegen konnten, z. B.). In diesem Fall hat `x-amz-date` Vorrang vor dem `Date`-Header. Aus diesem Grund muss der Datumseintrag in der Signatur den Wert des `x-amz-date`-Headers enthalten.

Hochladen

Dieses Beispiel lädt ein Objekt in einen virtuell gehosteten Bucket mit CNAME-Stil mit Metadaten.

Anforderung	StringToSign
`PUT /db-backup.dat.gz HTTP/1.1` `User-Agent: curl/7.15.5` `Host: static.example.com:8080` `Date: Tue, 27 Mar 2007 21:06:08 +0000` `x-amz-acl: public-read` `content-type: application/x-download` `Content-MD5: 4gJE4saaMU4BqNR0kLY+lw==` `X-Amz-Meta-ReviewedBy: joe@example.com` `X-Amz-Meta-ReviewedBy: jane@example.com` `X-Amz-Meta-FileChecksum: 0x02661779` `X-Amz-Meta-ChecksumAlgorithm: crc32` `Content-Disposition: attachment; filename=database.dat`	`PUT\n` `4gJE4saaMU4BqNR0kLY+lw==\n` `application/x-download\n` `Tue, 27 Mar 2007 21:06:08 +0000\n` `x-amz-acl:public-read\n` `x-amz-meta-checksumalgorithm:crc32\n` `x-amz-meta-filechecksum:0x02661779\n` `x-amz-meta-reviewedby:joe@example.com,jane@example.com\n`

Anforderung	StringToSign
`Content-Encoding: gzip` `Content-Length: 5913339` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE:`* *`jtBQa0Aq+DkULFI8qrpwIjGEx0E=`*	`/static.example.com/db-backup.dat` `.gz`

Beachten Sie, wie die „x-amz-“-Header sortiert, von Leerzeichen befreit und in Kleinbuchstaben umgewandelt wurden. Beachten Sie außerdem, dass mehrere Header mit demselben Namen unter Verwendung von Kommas zum Trennen der Werte verknüpft wurden.

Beachten Sie, wie nur die HTTP-Header `Content-Type` und `Content-MD5` in `StringToSign` erscheinen. Die anderen `Content-*`-Header erscheinen nicht.

Beachten Sie ebenfalls, dass`CanonicalizedResource` den Bucket-Namen beinhaltet, die HTTP Anforderungs-URI dagegen nicht. (Der Bucket wird vom Host-Header spezifiziert.)

Auflisten aller meiner Buckets

Anforderung	StringToSign
`GET / HTTP/1.1` `Host: s3.us-west-1.amazonaws.com` `Date: Wed, 28 Mar 2007 01:29:59 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE:qGdzdE`* *`RIC03wnaRNKh6OqZehG9s=`*	`GET\n` `\n` `\n` `Wed, 28 Mar 2007 01:29:59` `+0000\n` `/`

Unicode-Schlüssel

Anforderung	StringToSign
`GET /dictionary/fran%C3%A7ais/pr` `%c3%a9f%c3%a8re HTTP/1.1` `Host: s3.us-west-1.amazonaws.com`	`GET\n` `\n` `\n`

Anforderung	StringToSign
`Date: Wed, 28 Mar 2007 01:49:49 +0000` `Authorization: AWS AKIAIOSFODNN7EXAMPLE:DNEZGsoieTZ92F3bUfSPQcbGmlM=`	`Wed, 28 Mar 2007 01:49:49 +0000\n` `/dictionary/fran%C3%A7ais/pr%c3%a9f%c3%a8re`

> **Note**
>
> Die Elemente in `StringToSign`, die von der Anforderungs-URI abgeleitet wurden, werden unverändert übernommen, einschließlich der URL-Codierung und der Großschreibung.

Probleme mit dem Signieren von REST-Anforderungen

Wenn die Authentifizierung einer REST-Anforderung fehlschlägt, reagiert das System mit einem XML-Fehlerdokument auf die Anforderung. Die in diesem Fehlerdokument enthaltene Information soll Entwicklern helfen, das Problem zu diagnostizieren. Insbesondere erkennen Sie an dem `StringToSign`-Element des `SignatureDoesNotMatch`-Fehlerdokuments genau, welche Anforderungs-Kanonisierung das System verwendet.

Einige Toolkits fügen stillschweigend Header ein, die Sie zuvor nicht kennen, wie beispielsweise den Header `Content-Type` bei einem PUT. In den meisten dieser Fälle bleibt der Wert des eingefügten Headers konstant, sodass Sie fehlende Header unter Verwendung von Tools wie Ethereal oder tcpmon erkennen können.

Alternative zur Authentifizierung der Abfragezeichenfolge

Einige Anforderungstypen können Sie authentifizieren, indem Sie die angeforderten Informationen als Abfragezeichenfolgenparameter übergeben, statt den HTTP-Header `Authorization` zu verwenden. Dies ist praktisch, um direkten Zugriff durch einen Browser von Dritten auf Ihre privaten Amazon-S3-Daten zu ermöglichen, ohne dass die Anfrage über einen Proxy gesendet wird. Die Idee besteht in der Konstruierung einer „vorsignierten“ Anforderung und ihrer Codierung als einer URL, die vom Browser eines Endbenutzers geladen werden kann. Darüber hinaus können Sie eine vorsignierte Anforderung durch die Angabe einer Ablaufzeit begrenzen.

Weitere Informationen zur Verwendung von Abfrageparametern zum Authentifizieren von Anforderungn finden Sie unter Authentifizieren von Anforderungn: Verwenden von Abfrageparametern (AWS Signature Version 4) in der Amazon Simple Storage Service API-Referenz.

Beispiele für die Verwendung von AWS SDKs zum Generieren vorsignierter URLs finden Sie unter Gemeinsame Nutzung von Objekten mit vorsignierten URLs.

Erstellen einer Signatur

Das folgende Beispiel zeigt eine mit Abfragezeichenfolge authentifizierte Amazon-S3-REST-Anfrage.

```
GET /photos/puppy.jpg
?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Expires=1141889120&Signature=vjbyPxybdZaNmGa
%2ByT272YEAiv4%3D HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
Date: Mon, 26 Mar 2007 19:37:58 +0000
```

Die Authentifizierungsmethode für die Abfragezeichenkette benötigt keine spezifischen HTTP-Header. Stattdessen werden die erforderlichen Authentifizierungselemente als Abfragezeichenfolgenparameter angegeben:

Abfragezeichenfolgen-Parametername	Beispielwert	Beschreibung
`AWSAccessKeyId`	`AKIAIOSFODNN7EXAMPLE`	Ihre AWS-Zugriffsschlüssel-ID Gibt den geheimen AWS-Zugriffsschlüssel an, der verwendet wurde, um die Anforderung zu signieren, und somit indirekt die Identität des Entwicklers, der die Anforderung gestellt hat.
`Expires`	`1141889120`	Die Zeit, wann die Signatur abläuft, angegeben als die Anzahl der Sekunden, die seit dem 1. Januar 1970 00:00:00 UTC verstrichen sind. Eine Anforderung, die nach dieser Zeit empfangen wird (abhängig vom Server), wird abgelehnt.

Abfrageze ichenfolgen-Parametername	Beispielwert	Beschreibung
`Signature`	`vjbyPxybdZaNmGa%2B yT272YEAiv4%3D`	Die URL-Kodierung der Base64-Kodierung des HMAC-SHA1 von StringToSign.

Die Methode zur Authentifizierung der Abfragezeichenfolge unterscheidet sich leicht von der üblichen Methode, aber nur im Format des `Signature`-Anforderungsparameters und des `StringToSign`-Elements. Die folgende Pseudogrammatik verdeutlicht die Authentifizierungsmethode für die Abfragezeichenfolge.

```
Signature = URL-Encode( Base64( HMAC-SHA1( YourSecretAccessKey, UTF-8-Encoding-
Of( StringToSign ) ) ) );

StringToSign = HTTP-VERB + "\n" +
    Content-MD5 + "\n" +
    Content-Type + "\n" +
    Expires + "\n" +
    CanonicalizedAmzHeaders +
    CanonicalizedResource;
```

`YourSecretAccessKey` ist die ID des geheimen AWS-Zugriffsschlüssels, die Ihnen Amazon zuordnet, wenn Sie sich als Amazon-Web-Services-Entwickler anmelden. Beachten Sie, wie die `Signature` URL-codiert wird, damit sie für die Platzierung im Abfragestring geeignet ist. Beachten Sie auch, dass in `StringToSign` das HTTP-Positionselement `Date` durch `Expires` ersetzt wurde. `CanonicalizedAmzHeaders` und `CanonicalizedResource` sind gleich.

> **Note**
>
> In der Methode für die Abfrage-String-Authentifizierung verwenden Sie den Header `Date` oder `x-amz-date request` nicht, wenn Sie die zu signierende Zeichenfolge berechnen.

Authentifizierung von Abfragezeichenfolgen-Anforderungen

Anforderung	StringToSign
`GET /photos/puppy.jpg?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&` `    Signature=NpgCjnDzrM%2BWFzoENXmpNDUsSn8%3D&` `    Expires=1175139620 HTTP/1.1` `Host: awsexamplebucket1.s3.us-west-1.amazonaws.com`	`GET\n` `\n` `\n` `1175139620\n` `/awsexamplebucket1/photos/puppy.jpg`

Wir gehen davon aus, dass ein Browser bei einer GET-Anforderung keinen Content-MD5- oder Content-Type-Header bereitstellt, und auch keine x-amz--Header einrichten, diese Teile von `StringToSign` bleiben deshalb leer.

Verwendung der Base64-Codierung

Signaturen von HMAC-Anforderungen müssen mit Base64 codiert werden. Die Base64-Codierung wandelt die Signatur in einen einfachen ASCII-String um, der der Anforderung hinzugefügt werden kann. Zeichen, die in der Signatur erscheinen können, wie beispielsweise Plus (+), Schrägstrich (/) oder Gleichheitszeichen (=), müssen wie in einer URI codiert werden. Enthält beispielsweise der Authentifizierungscode ein Plussymbol (+), codieren Sie es in der Anforderung als &2B. Einen Schrägstrich codieren Sie als &2F, ein Gleichheitszeichen als %3D.

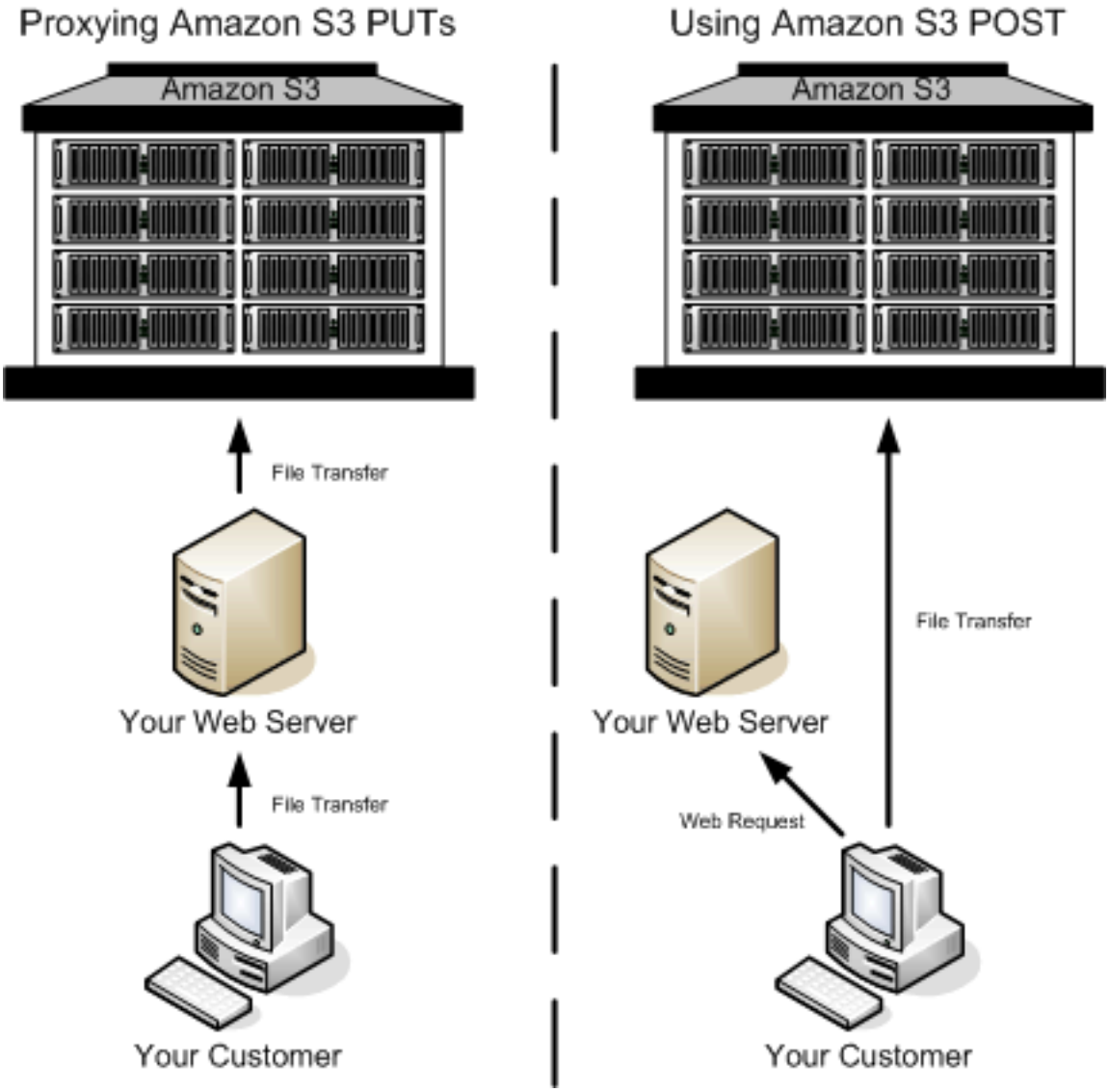
Beispiele für die Base64-Codierung finden Sie in Amazon S3 Authentifizierungsbeispiele.

Browserbasierte Uploads mit POST (AWS Signature Version 2)

Amazon S3 unterstützt POST; damit können Ihre Benutzer Inhalte direkt zu Amazon S3 hochladen. POST dient dazu, Upload-Vorgänge zu vereinfachen, die Latenz bei Uploads zu reduzieren und Ihnen bei Anwendungen Geld zu sparen, bei denen Benutzer Daten zur Speicherung auf Amazon S3 hochladen.

> **Note**
>
> Die in diesem Abschnitt besprochene Anforderungs-Authentifizierung basiert auf AWS Signature Version 2, einem Protokoll für die Authentifizierung eingehender API-Anforderungen an AWS-Services.
> Amazon S3 unterstützt jetzt Signature Version 4, ein Protokoll für die Authentifizierung eingehender API-Anforderungn an AWS-Services, in allen AWS-Regionen. Derzeit unterstützen vor dem 30. Januar 2014 erstellte AWS-Regionen weiterhin das vorherige Protokoll, Signature Version 2. Alle nach dem 30. Januar 2014 neu erstellten Regionen unterstützen nur Signature Version 4, weshalb alle Anforderungn an diese Regionen Signature Version 4 verwenden müssen. Weitere Informationen finden Sie unter Authenticating Requests in Browser-Based Uploads Using POST (Authentifizieren von Anforderungn in browserbasierten Uploads mit POST) (AWS Signature Version 4) in der API-Referenz für Amazon Simple Storage Service.

Die folgende Abbildung zeigt einen Upload mit Amazon S3 POST.



Upload mit POST

1	Der Benutzer öffnet einen Webbrowser, und greift auf Ihre Webseite zu.
2	Ihre Webseite enthält ein HTTP-Formular mit allen Informationen, die der Benutzer benötigt, um Inhalte zu Amazon S3 hochladen zu können.
3	Der Benutzer lädt die Inhalte direkt zu Amazon S3 hoch.

> **Note**
>
> Die Abfrage-String-Authentifizierung wird für POST nicht unterstützt.

HTML-Formulare (AWS Signature Version 2)

Themen

Wenn Sie mit Amazon S3 kommunizieren, verwenden Sie normalerweise die REST- oder SOAP-API zum Laden, Abrufen, Löschen und für andere Operationen. Mit POST laden die Benutzer die Daten direkt zu Amazon S3 über ihre Browser hoch, die nicht in der Lage sind, die SOAP-API zu verarbeiten oder eine REST-PUT-Anfrage zu erstellen.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, SOAP steht über HTTPS aber noch zur Verfügung. Neue Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, statt SOAP entweder die REST-API oder die AWS-SDKs zu verwenden.

Um Benutzern zu erlauben, Inhalte über ihre Browser zu Amazon S3 hochzuladen, verwenden Sie HTML-Formulare. HTML-Formulare bestehen aus einer Formulardeklaration und Formularfeldern. Die Formulardeklaration enthält allgemeine Informationen zu der Anfrage. Die Formularfelder enthalten detaillierte Informationen zu der Anfragen sowie die Richtlinie, die verwendet wird, um die Anfrage zu authentifizieren und um sicherzustellen, dass sie den von Ihnen angegebenen Bedingungen entspricht.

> **Note**
>
> Daten und Grenzen des Formulars (ausschließlich der Inhalte der Datei) dürfen 20 KB nicht überschreiten.

Dieser Abschnitt erläutert die Verwendung von HTML-Formularen.

HTML-Formular-Kodierung

Das Formular und die Richtlinie müssen gemäß UTF-8 kodiert sein. Sie können die UTF-8-Kodierung für das Formular anwenden, indem Sie sie im HTML-Kopf oder als Anfragekopf angeben.

> **Note**
>
> Die HTML-Formulardeklaration akzeptiert keine Abfrage-String-Authentifizierungsparameter.

Nachfolgend sehen Sie ein Beispiel für die UTF-8-Kodierung im HTML-Kopf:

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
```

Nachfolgend sehen Sie ein Beispiel für die UTF-8-Kodierung im Anfragekopf:

```
Content-Type: text/html; charset=UTF-8
```

HTML-Formulardeklaration

Die Formulardeklaration besteht aus drei Komponenten: der Aktion, der Methode und dem Umschließungstyp. Wenn einer dieser Werte nicht korrekt eingestellt ist, schlägt die Anforderung fehl.

Die Aktion gibt die URL an, die die Anforderung verarbeitet; dies muss die URL des Buckets sein. Wenn der Name Ihres Buckets beispielsweise `awsexamplebucket1` lautet und die Region USA West (Nordkalifornien) ist, lautet die URL `https://awsexamplebucket1.s3.us-west-1.amazonaws.com/`.

> **Note**
>
> Der Schlüsselname wird in einem Formularfeld angegeben.

Die Methode muss POST sein.

Der Umschließungstyp (enctype) muss angegeben werden und auf Multipart/Form-Daten für Datei- und Textbereich-Uploads gesetzt sein. Weitere Informationen finden Sie unter RFC 1867.

Example

Das folgende Beispiel zeigt eine Formulardeklaration für den Bucket „awsexamplebucket1“.

```
<form action="https://awsexamplebucket1.s3.us-west-1.amazonaws.com/" method="post"

enctype="multipart/form-data">
```

HTML-Formularfelder

Die folgende Tabelle beschreibt die Felder, die in einem HTML-Formular verwendet werden können.

> **Note**
>
> Die Variable `${filename}` wird automatisch durch den Namen der Datei ersetzt, den der Benutzer bereitstellt, und der von allen Formularfeldern anerkannt wird. Wenn der Browser oder Client einen vollständigen oder teilweisen Pfad zu der Datei bereitstellt, wird nur der Text nach dem letzten Schrägstrich (/) oder umgekehrten Schrägstrich (\) verwendet. Beispielsweise wird „C:\Program Files\directory1\file.txt“ als „file.txt“ interpretiert. Wenn keine Datei bzw. kein Dateiname angegeben ist, wird die Variable durch eine leere Zeichenfolge ersetzt.

Feldname	Beschreibung	Erforderlich
`AWSAccessKeyId`	Die AWS-Zugriffsschlüssel-ID des Besitzers des Buckets, der den anonymen Benutzerz ugriff für eine Anforderung gewährt, die den Einschränkungen in der Richtlinie entspricht. Dies ist ein Pflichtfeld, wenn die Anforderung ein Richtliniendokument beinhaltet.	Bedingt
`acl`		Nein

Feldname	Beschreibung	Erforderlich
	Eine Amazon-S3-Zugriffskontrollliste (ACL). Wenn eine ungültige Zugriffskontrollliste angegeben ist, wird ein Fehler generiert. Weitere Informationen zu ACLs finden Sie unter Zugriffssteuerungslisten (ACLs). Typ: Zeichenfolge Standard: privat Zulässige Werte: `private | public-read | public-read-write | aws-exec-read | authenticated-read | bucket-owner-read | bucket-owner-full-control`	
`Cache-Control, Content-Type, Content-Disposition, Content-Encoding, Expires`	REST-spezifische Köpfe. Weitere Informationen finden Sie unter PUT Object.	Nein
`key`	Der Name des hochgeladenen Schlüssels. verwenden Sie die Variable $(filename), um den von dem Benutzer angegebenen Dateinamen zu verwenden. Zum Beispiel: Wenn Benutzerin „Betty“ die Datei „lolcatz.jpg“ hochlädt, und Sie „/user/betty/${filename}“ angeben, wird die Datei als „/user/betty/lolcatz.jpg“ gespeichert. Weitere Informationen finden Sie unter Arbeiten mit Objekt-Metadaten.	Ja

Feldname	Beschreibung	Erforderlich
`policy`	Die Sicherheitsrichtlinie, die angibt, was in der Anforderung zulässig ist. Anforderungen ohne Sicherheitsrichtlinie gelten als anonym und sind nur auf öffentlich zugänglichen Buckets erfolgreich.	Nein
`success_action_redirect, redirect`	Die URL, zu der der Client nach einem erfolgreichen Upload umgeleitet wird. Amazon S3 hängt die Bucket-, Schlüssel- und Etag-Werte als Abfragezeichenfolgenparameter an die URL an. Wenn success_action_redirect nicht angegeben ist, gibt Amazon S3 den leeren Dokumententyp zurück, der im Feld success_action_status angegeben ist. Wenn Amazon S3 die URL nicht interpretieren kann, wird das Feld ignoriert. Wenn der Upload fehlschlägt, zeigt Amazon S3 einen Fehler an und leitet den Benutzer nicht zu einer URL um. Weitere Informationen finden Sie unter Umleitung. **Note** Der Name des Umleitungsfeldes ist veraltet, und das Feld wird künftig nicht mehr unterstützt.	Nein

Feldname	Beschreibung	Erforderlich
`success_action_status`	Der an den Client bei einem erfolgreichen Upload ausgegebene Statuscode, wenn success_action_redirect nicht angegeben ist. Gültige Werte sind 200, 201 und 204 (Standard). Wenn der Wert auf 200 oder 204 gesetzt ist, gibt Amazon S3 ein leeres Dokument mit dem Statuscode 200 oder 204 aus. Wenn der Wert auf 201 gesetzt ist, gibt Amazon S3 ein XML-Dokument mit dem Statuscode 201 aus. Informationen zum Inhalt des XML-Dokuments finden Sie unter POST Object. Wenn der Wert nicht oder auf einen falschen Wert gesetzt ist, gibt Amazon S3 ein leeres Dokument mit dem Statuscode 204 aus. **Note** Einige Versionen von Adobe Flash Player können HTTP-Antworten ohne Text nicht korrekt bearbeiten. Zur Unterstützung von Uploads über Adobe Flash empfehlen wir, `success_action_status` auf 201 zu setzen.	Nein

Feldname	Beschreibung	Erforderlich
`signature`	Die HMAC-Signatur, die mithilfe des geheimen Zugriffsschlüssels konstruiert wird, der der angegebenen AWSAccessKeyId entspricht. Dies ist ein Pflichtfeld, wenn die Anforderung ein Richtliniendokument beinhaltet. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3.	Bedingt
`x-amz-security-token`	Ein Sicherheitstoken, das von Sitzungsanmeldeinformationen verwendet wird Wenn die Anfrage Amazon DevPay verwendet, benötigt sie zwei `x-amz-security-token` -Formularfelder: eins für das Produkttoken und eins für das Benutzertoken. Wenn die Anforderung Sitzungsanmeldedaten verwendet, ist ein `x-amz-security-token` -Formular erforderlich. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch.	Nein
Andere Feldnamen mit dem Präfix x-amz-meta-	Vom Benutzer angegebene Metadaten Amazon S3 validiert oder verwendet diese Daten nicht. Weitere Informationen finden Sie unter PUT Object.	Nein

Feldname	Beschreibung	Erforderlich
file	Datei oder Textinhalt. Die Datei oder der Inhalt muss das letzte Feld des Formulars sein. Alle Felder darunter werden ignoriert. Sie können nicht mehr als eine Datei zur gleichen Zeit hochladen.	Ja

Richtlinienerstellung

Themen

- Ablauf
- Bedingungen
- Übereinstimmung von Bedingungen
- Escape-Zeichen

Die Richtlinie ist ein in UTF-8 und Base64 kodiertes JSON-Dokument, das die Bedingungen angibt, die die Anforderungen erfüllen muss; sie wird zur Authentifizierung des Inhalts verwendet. Je nach dem, wie Sie Ihre Richtliniendokumente gestalten, können Sie sie pro Upload, pro Benutzer, für alle Uploads oder nach anderen Kriterien, die Ihren Anforderungen entsprechen, verwenden.

> **Note**
>
> Obwohl das Richtliniendokument optional ist, empfehlen nachdrücklich, es zu verwenden, anstatt einen Bucket öffentlich beschreibbar zu machen.

Nachstehend finden Sie ein Beispiel für ein Richtliniendokument:

```
{ "expiration": "2007-12-01T12:00:00.000Z",

  "conditions": [
```

```
    {"acl": "public-read" },

    {"bucket": "awsexamplebucket1" },

    ["starts-with", "$key", "user/eric/"],

  ]

}
```

Das Richtliniendokument enthält den Ablauf und die Bedingungen.

Ablauf

Das Ablaufelement gibt das Ablaufdatum der Richtlinie im Datumsformat nach ISO 8601 UTC an. Beispielsweise bedeutet „2007-12-01T12:00:00.000Z“, dass die Richtlinie ab Mitternacht (UTC) am 01. 12. 2007 ungültig ist. Die Angabe des Ablaufs ist in einer Richtlinie erforderlich.

Bedingungen

Die Bedingungen in dem Richtliniendokument validieren den Inhalt des hochgeladenen Objekts. Jedes Formularfeld, das Sie in dem Formular angeben (ausgenommen AWSAccessKeyId, Signatur, Datei, Richtlinie und Feldnamen mit dem Präfix x-ignore-) muss in der Liste der Bedingungen aufgeführt werden.

> **Note**
>
> Wenn Sie mehrere Felder mit dem gleichen Namen haben, müssen die Werte durch Kommata abgeteilt sein. Zum Beispiel: Wenn Sie zwei Felder mit der Bezeichnung „x-amz-meta-tag“ haben und das erste den Wert „Ninja“ und das zweite den Wert „Stallman“ hat, setzen Sie das Richtliniendokument auf `Ninja,Stallman`.
> Alle Variablen in dem Formular werden vor der Validierung der Richtlinie erweitert. Daher müssen alle Bedingungsabgleiche anhand der erweiterten Felder vorgenommen werden. Zum Beispiel: Wenn Sie das Schlüsselfeld auf `user/betty/${filename}` setzen, ist Ihre Richtlinie möglicherweise `[ "starts-with", "$key", "user/betty/" ]`. Geben Sie nicht ei `[ "starts-with", "$key", "user/betty/${filename}" ]`. Weitere Informationen finden Sie unter Übereinstimmung von Bedingungen.

Die folgende Tabelle beschreibt die Bedingungen für Richtliniendokumente.

Elementname	Beschreibung
acl	Gibt die Bedingungen an, die die ACL erfüllen muss. Unterstützt exakte Übereinstimmung und `starts-with` .
content-length-range	Gibt die erlaubte Mindest- und Höchstgröße des hochgelad enen Inhalts an. Unterstützt den Bereichsabgleich.
Cache-Control, Content-Type, Content-Disposition, Content-E ncoding, Expires	REST-spezifische Köpfe. Unterstützt exakte Übereinstimmung und `starts-with` .
Schlüssel	Der Name des hochgeladenen Schlüssels. Unterstützt exakte Übereinstimmung und `starts-with` .
success_action_redirect, redirect	Die URL, zu der der Client nach einem erfolgreichen Upload umgeleitet wird. Unterstützt exakte Übereinstimmung und `starts-with` .
success_action_status	Der an den Client bei einem erfolgreichen Upload ausgegebe ne Statuscode, wenn success_action_redirect nicht angegeben ist. Unterstützt exakte Übereinstimmung.
x-amz-security-token	Amazon DevPay-Sicherheitstoken. Jede Anfrage, die Amazon DevPay verwendet, erfordert zwei `x-amz-security-token` -Formularfelder erforderl ich: eins für das Produkttoken und eins für das Benutzert oken. Daher müssen die Werte durch Kommata voneinand

Elementname	Beschreibung
	er getrennt werden. Zum Beispiel: Wenn der Benutzer-Token `eW91dHViZQ==` und der Produkt-Token `b0hnNVNKWVJIQTA=` ist, setzen Sie den Richtlinieneintrag auf: `{ "x-amz-security-token": "eW91dHViZQ==,b0hnNVNKWVJIQTA=" }` .
Andere Feldnamen mit dem Präfix x-amz-meta-	Vom Benutzer angegebene Metadaten Unterstützt exakte Übereinstimmung und `starts-with` .

> **Note**
>
> Wenn Ihr Toolkit weitere Felder hinzufügt (beispielsweise fügt Flash den Dateinamen hinzu), müssen Sie diese dem Richtliniendokument hinzufügen. Wenn Sie diese Funktionalität steuern können, setzen Sie `x-ignore-` vor das Feld, sodass Amazon S3 die Funktion ignoriert und künftige Versionen davon unbeeinflusst lässt.

Übereinstimmung von Bedingungen

Die folgende Tabelle beschreibt die Übereinstimmungstypen von Bedingungen. Obwohl Sie für jedes Formularfeld, das Sie in dem Formular angeben, eine Bedingung angeben müssen, können Sie auch komplexere Übereinstimmungskriterien erstellen, indem Sie für ein Formularfeld mehrere Bedingungen angeben.

Bedingung	Beschreibung
Genaue Übereinstimmung	Hierbei müssen die Felder spezifische Werte enthalten. In diesem Beispiel ist angegeben, dass die ACL auf „public-read“ gesetzt werden muss: `{"acl": "public-read" }` Dieses Beispiel zeigt eine alternative Möglichkeit, um anzuzeigen, dass die ACL auf „public-read“ gesetzt werden muss:

Bedingung	Beschreibung
	`[ "eq", "$acl", "public-read" ]`
Beginnt mit	Verwenden Sie diese Bedingung, wenn der Wert mit einem bestimmten Wert beginnen muss. In diesem Beispiel wird angegeben, dass der Schlüssel mit „user/betty“ beginnen muss: `["starts-with", "$key", "user/betty/"]`
Übereinstimmung mit beliebigem Inhalt	Verwenden Sie „starts-with“ mit einem leeren Wert, um die Richtlinie so zu konfigurieren, dass in einem Feld beliebiger Inhalt zulässig ist. Dieses Beispiel lässt jeden Wert für success_action_redirect zu: `["starts-with", "$success_action_redirect", ""]`
Angabe von Bereichen	Wenn Felder Bereiche akzeptieren, trennen Sie die Ober- und die Untergrenze durch ein Komma voneinander ab. In diesem Beispiel sind Dateigrößen von 1 bis 10 Megabyte erlaubt: `["content-length-range", 1048579, 10485760]`

Escape-Zeichen

Die folgende Tabelle beschreibt Zeichen, für die in einem Richtliniendokument ein Escape-Zeichen verwendet werden muss.

Escape-Sequenz	Beschreibung
\\	Umgekehrter Schrägstrich

Escape-Sequenz	Beschreibung
\$	Dollarzeichen
\b	Backspace
\f	Seitenvorschub
\n	Neue Zeile
\r	Zeilenumschaltung
\t	Horizontaler Tabulator
\V	Vertikaler Tabulator
\u*xxxx*	Alle Unicode-Zeichen

Erstellen einer Signatur

Schritt	Beschreibung
1	Kodieren Sie die Richtlinie mit UTF-8.
2	Kodieren Sie diese UTF-8-Bytes mit Base64.
3	Signieren Sie die Richtlinie mit Ihrem geheimen Zugriffsschlüssel unter Verwendung von HMAC SHA-1.
4	Kodieren Sie die SHA-1-Signatur mit Base64.

Allgemeine Informationen über die Authentifizierung finden Sie unter Identity and Access Management in Amazon S3.

Umleitung

In diesem Abschnitt wird beschrieben, wie Sie mit Umleitungen umgehen.

Allgemeine Umleitung

Beim Abschluss der POST-Anforderung wird der Benutzer zu dem Ort umgeleitet, den Sie im Feld `success_action_redirect` angeben. Wenn Amazon S3 die URL nicht interpretieren kann, wird das Feld `success_action_redirect` ignoriert.

Wenn `success_action_redirect` nicht angegeben ist, gibt Amazon S3 den leeren Dokumententyp zurück, der im Feld `success_action_status` angegeben ist.

Wenn die POST-Anfrage fehlschlägt, zeigt Amazon S3 einen Fehler an und führt keine Umleitung durch.

Umleitung vor dem Upload

Wenn Ihr Bucket mit <CreateBucketConfiguration> erstellt wurde, benötigen Ihre Endbenutzer möglicherweise eine Umleitung. Wenn dies der Fall ist, kann es sein, das einige Browser mit der Umleitung nicht korrekt umgehen. Dies kommt relativ selten vor, meistens dann, wenn der Bucket gerade eben erstellt wurde.

Upload-Beispiele (AWS Signature Version 2)

Themen

- Datei-Upload
- Textbereich-Upload

> **Note**
>
> Die in diesem Abschnitt besprochene Anforderungs-Authentifizierung basiert auf AWS Signature Version 2, einem Protokoll für die Authentifizierung eingehender API-Anforderungen an AWS-Services.
> Amazon S3 unterstützt jetzt Signature Version 4, ein Protokoll für die Authentifizierung eingehender API-Anforderungn an AWS-Services, in allen AWS-Regionen. Derzeit

unterstützen vor dem 30. Januar 2014 erstellte AWS-Regionen weiterhin das vorherige Protokoll, Signature Version 2. Alle nach dem 30. Januar 2014 neu erstellten Regionen unterstützen nur Signature Version 4, weshalb alle Anforderungn an diese Regionen Signature Version 4 verwenden müssen. Weitere Informationen finden Sie unter Beispiele: Browserbasiertes Hochladen mit HTTP POST (Mit AWS Signature Version 4) in der API-Referenz für Amazon Simple Storage Service.

Datei-Upload

Dieses Beispiel zeigt den vollständigen Vorgang der Erstellung einer Richtlinie und eines Formulars für den Upload eines Dateianhangs.

Erstellung von Richtlinie und Formular

Die folgende Richtlinie unterstützt Uploads zu Amazon S3 für den Bucket „awsexamplebucket1“.

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "awsexamplebucket1"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
    {"success_action_redirect": "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
successful_upload.html"},
    ["starts-with", "$Content-Type", "image/"],
    {"x-amz-meta-uuid": "14365123651274"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}
```

Für diese Richtlinie ist Folgendes erforderlich:

- Der Upload muss vor 12:00 Uhr UTC am 1. Dezember 2007 erfolgen.
- Der Inhalt muss zum Bucket „awsexamplebucket1“ hochgeladen werden.
- Der Schlüssel muss mit „user/eric/“ beginnen.
- Die ACL ist auf „public-read“ gesetzt.
- Die URL für success_action_redirect wird auf https://awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html festgelegt.
- Bei dem Objekt handelt es sich um eine Bilddatei.

- Der Tag x-amz-meta-uuid muss auf 14365123651274 gesetzt sein.
- Der Tag x-amz-meta-tag kann einen beliebigen Wert enthalten.

Nachfolgend sehen Sie eine mit Base64 kodierte Version dieser Richtlinie.

```
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN
```

Erstellen Sie unter Verwendung Ihrer Anmeldedaten eine Signatur, zum Beispiel ist `0RavWzkygo6QX9caELEqKi9kDbU=` die Signatur für das vorangehende Richtliniendokument.

Das folgende Formular unterstützt eine POST-Anforderung an den Bucket DOC-EXAMPLE-BUCKET, der diese Richtlinie verwendet.

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
  ...
  <form action="https://DOC-EXAMPLE-BUCKET.s3.us-west-1.amazonaws.com/" method="post"
 enctype="multipart/form-data">
    Key to upload: <input type="input" name="key" value="user/eric/" /><br />
    <input type="hidden" name="acl" value="public-read" />
    <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html" />
    Content-Type: <input type="input" name="Content-Type" value="image/jpeg" /><br />
    <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
    Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
    <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
    <input type="hidden" name="Policy" value="POLICY" />
    <input type="hidden" name="Signature" value="SIGNATURE" />
    File: <input type="file" name="file" /> <br />
    <!-- The elements after this will be ignored -->
    <input type="submit" name="submit" value="Upload to Amazon S3" />
  </form>
  ...
</html>
```

Beispielanforderung

Diese Anforderung geht davon aus, dass das hochgeladene Bild 117.108 Bytes groß ist, die Bilddaten nicht eingeschlossen.

```
POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
 Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: 118698

--9431149156168
Content-Disposition: form-data; name="key"

user/eric/MyPicture.jpg
--9431149156168
Content-Disposition: form-data; name="acl"

public-read
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html
--9431149156168
Content-Disposition: form-data; name="Content-Type"

image/jpeg
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-tag"

Some,Tag,For,Picture
--9431149156168
```

```
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--9431149156168
Content-Disposition: form-data; name="Policy"

eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN
--9431149156168
Content-Disposition: form-data; name="Signature"

0RavWzkygo6QX9caELEqKi9kDbU=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg

...file content...
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--9431149156168--
```

Beispielantwort

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
successful_upload.html?bucket=awsexamplebucket1&key=user/eric/
MyPicture.jpg&etag=&quot;39d459dfbc0faabbb5e179358dfb94c3&quot;
Server: AmazonS3
```

Textbereich-Upload

Themen

- Erstellung von Richtlinie und Formular
- Beispielanforderung
- Beispielantwort

Das folgende Beispiel zeigt den vollständigen Prozess der Erstellung einer Richtlinie und eines Formulars für den Upload eines Textbereiches. Der Upload eines Textbereiches ist nützlich für die Übermittlung von Benutzern erstellter Inhalte, etwa von Blog-Postings.

Erstellung von Richtlinie und Formular

Die folgende Richtlinie unterstützt das Hochladen von Textbereichen zu Amazon S3 für den Bucket „awsexamplebucket1“.

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "awsexamplebucket1"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
    {"success_action_redirect": "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
new_post.html"},
    ["eq", "$Content-Type", "text/html"],
    {"x-amz-meta-uuid": "14365123651274"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}
```

Für diese Richtlinie ist Folgendes erforderlich:

- Der Upload muss vor 12:00 Uhr GMT am 1. Dezember 2007 erfolgen.
- Der Inhalt muss zum Bucket „awsexamplebucket1“ hochgeladen werden.
- Der Schlüssel muss mit „user/eric/“ beginnen.
- Die ACL ist auf „public-read“ gesetzt.
- Die URL für success_action_redirect ist auf „https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html“ festgelegt.
- Bei dem Objekt handelt es sich um HTML-Text.
- Der Tag x-amz-meta-uuid muss auf 14365123651274 gesetzt sein.
- Der Tag x-amz-meta-tag kann einen beliebigen Wert enthalten.

Nachfolgend sehen Sie eine mit Base64 kodierte Version dieser Richtlinie.

```
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXR
```

```
pb25zIjogWwogICAgeyJidWNrZXQiOiAiam9obnNtaXRoIn0sCiAgICBbInN0YXJ0cy13aXRoIiwgIiRrZXkiLCAidXNlci
LAogICAgeyJhY2wiOiAicHVibGljLXJlYWQifSwKICAgIHsic3VjY2Vzc19hY3Rpb25fcmVkaXJlY3QiOiAiaHR0cDovL2p
C5zMy5hbWF6b25hd3MuY29tL25ld19wb3N0Lmh0bWwifSwKICAgIFsiZXEiLCAiJENvbnRlbnQtVHlwZSIsICJ0ZXh0L2h0
CAgIHsieC1hbXotbWV0YS11dWlkIjogIjE0MzY1MTIzNjUxMjc0In0sCiAgICBbInN0YXJ0cy13aXRoIiwgIiR4LWFtei1t
IsICIiXQogIF0KfQo=
```

Erstellen Sie unter Verwendung Ihrer Anmeldedaten eine Signatur. Beispielsweise ist `qA7FWXKq6VvU68lI9KdveT1cWgF=` die Signatur für das vorherige Richtliniendokument.

Das folgende Formular unterstützt eine POST-Anforderung an den Bucket DOC-EXAMPLE-BUCKET, der diese Richtlinie verwendet.

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
  ...
  <form action="https://DOC-EXAMPLE-BUCKET.s3.us-west-1.amazonaws.com/" method="post"
 enctype="multipart/form-data">
    Key to upload: <input type="input" name="key" value="user/eric/" /><br />
    <input type="hidden" name="acl" value="public-read" />
    <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html" />
    <input type="hidden" name="Content-Type" value="text/html" />
    <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
    Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
    <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
    <input type="hidden" name="Policy" value="POLICY" />
    <input type="hidden" name="Signature" value="SIGNATURE" />
    Entry: <textarea name="file" cols="60" rows="10">

Your blog post goes here.

    </textarea><br />
    <!-- The elements after this will be ignored -->
    <input type="submit" name="submit" value="Upload to Amazon S3" />
  </form>
  ...
</html>
```

Beispielanforderung

Diese Anforderung geht davon aus, dass das hochgeladene Bild 117.108 Bytes groß ist, die Bilddaten nicht eingeschlossen.

```
POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
 Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=178521717625888
Content-Length: 118635

-178521717625888
Content-Disposition: form-data; name="key"

ser/eric/NewEntry.html
--178521717625888
Content-Disposition: form-data; name="acl"

public-read
--178521717625888
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html
--178521717625888
Content-Disposition: form-data; name="Content-Type"

text/html
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-tag"

Interesting Post
--178521717625888
```

```
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--178521717625888
Content-Disposition: form-data; name="Policy"
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN

--178521717625888
Content-Disposition: form-data; name="Signature"

qA7FWXKq6VvU68lI9KdveT1cWgF=
--178521717625888
Content-Disposition: form-data; name="file"

...content goes here...
--178521717625888
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--178521717625888--
```

Beispielantwort

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html?
bucket=awsexamplebucket1&key=user/eric/
NewEntry.html&etag=40c3271af26b7f1672e41b8a274d28d4
Server: AmazonS3
```

POST mit Adobe Flash

In diesem Abschnitt wird die Verwendung von POST mit Adobe Flash beschrieben.

Adobe Flash Player-Sicherheit

Standardmäßig verbietet das Sicherheitsmodell von Adobe Flash, dass Adobe Flash Players Netzwerkverbindungen zu Servern außerhalb der Domäne der SWF-Datei herstellen.

Um diesen Standard zu übergehen, müssen Sie eine öffentlich lesbare crossdomain.xml-Datei zu dem Bucket hochladen, der die POST-Uploads annehmen soll. Nachfolgend sehen Sie ein Beispiel für eine solche crossdomain.xml-Datei.

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.macromedia.com/xml/dtds/cross-domain-policy.dtd">
<cross-domain-policy>
<allow-access-from domain="*" secure="false" />
</cross-domain-policy>
```

> **Note**
>
> Weitere Informationen zum Sicherheitsmodell von Adobe Flash finden Sie auf der Adobe-Website.
> Durch die Hinzufügung der crossdomain.xml-Datei kann jeder Adobe Flash Player eine Verbindung zu der crossdomain.xml-Datei in Ihrem Bucket herstellen; diese Datei gewährt jedoch keinen Zugriff auf den Amazon-S3-Bucket selbst.

Überlegungen zu Adobe Flash

Die FileReference-API in Adobe Flash fügt das Formularfeld `Filename` zu der POST-Anforderung hinzu. Wenn Sie Adobe Flash-Anwendungen erstellen, die unter Verwendung der FileReference-API-Aktion Uploads zu Amazon S3 durchführen, fügen Sie die folgende Bedingung in Ihre Richtlinie ein:

```
['starts-with', '$Filename', '']
```

Einige Versionen von Adobe Flash Player können HTTP-Antworten ohne Text nicht korrekt bearbeiten. Um POST so zu konfigurieren, dass es eine Antwort ausgibt, die nicht leer ist, setzen Sie `success_action_status` auf 201. Amazon S3 gibt dann ein XML-Dokument mit dem Statuscode 201 zurück. Informationen zum Inhalt des XML-Dokuments finden Sie unter POST Object. Informationen zu den Formularfeldern finden Sie unter HTML-Formularfelder.

Bewährte Methoden für Designmuster: Optimieren der Leistung von Amazon S3

Ihre Anwendungen erreichen schnell Tausende von Transaktionen pro Sekunde bei der Anfrageleistung, wenn Speicherinhalte zu Amazon S3 hochgeladen oder von dort abgerufen werden. Amazon S3 wird automatisch auf hohe Anfrageraten skaliert. Ihre Anwendung kann beispielsweise mindestens 3 500 PUT/COPY/POST/DELETE- oder 5 500 GET/HEAD-Anforderungen pro Sekunde pro partitioniertem Amazon-S3-Präfix erreichen. Es gibt keine Einschränkungen für die Anzahl der Präfixe in einem Bucket. Sie können Ihre Lese- und Schreibleistung steigern, indem Sie Parallelisierung durchführen. Wenn Sie beispielsweise 10 Präfixe in einem Amazon S3-Bucket für parallele Lesevorgänge einrichten, können Sie damit die Leseleistung auf 55 000 Leseanfragen pro Sekunde skalieren. Auf ähnliche Weise können Sie Schreibvorgänge skalieren, indem Sie auf mehrere Präfixe schreiben. Die Skalierung erfolgt sowohl bei Lese- als auch bei Schreiboperationen schrittweise und nicht sofort. Während Amazon S3 auf Ihre neue höhere Anforderungsrate skaliert, können einige 503 (Slow Down)-Fehler auftreten. Diese Fehler werden nicht mehr angezeigt, sobald die Skalierung abgeschlossen ist. Weitere Informationen zum Erstellen und Verwenden von Präfixen finden Sie unter Organisieren von Objekten mit Präfixen.

So prüfen etwa manche Data Lake-Anwendungen in Amazon S3 Millionen oder Milliarden von Objekten auf Anfragen, die über Petabytes von Daten ausgeführt werden. Solche Data-Lake-Anwendungen erreichen Single-Instance-Übertragungsraten, die die Netzwerkschnittstellennutzung für ihre Amazon EC2-Instance maximieren und bis zu 100 GB/s auf einer einzelnen Instance erreichen können. Anschließend aggregieren diese Anwendungen den Durchsatz über mehrere Instances hinweg, um mehrere Terabit pro Sekunde zu erreichen.

Andere Anwendungen sind latenzempfindlich, wie etwa Social-Media-Messaginganwendungen. Diese Anwendungen können konsistente kleine Objektlatenzen (und first-byte-out Latenzen für größere Objekte) von etwa 100–200 Millisekunden erreichen.

Andere - AWS Services können auch dazu beitragen, die Leistung für verschiedene Anwendungsarchitekturen zu beschleunigen. Wenn Sie beispielsweise höhere Übertragungsraten über eine einzelne HTTP-Verbindung oder Latenzen im einstelligen Millisekundenbereich wünschen, verwenden Sie Amazon CloudFront oder Amazon ElastiCache für die Zwischenspeicherung mit Amazon S3.

Und wenn Sie eine schnelle Datenübertragung über große Distanzen zwischen einem Client und einem S3-Bucket wünschen, verwenden Sie Konfigurieren schneller, sicherer Dateiübertragungen

mit Amazon S3 Transfer Acceleration. Transfer Acceleration verwendet die global verteilten Edge-Standorte in CloudFront , um den Datentransport über geografische Entfernungen zu beschleunigen. Wenn Ihr Amazon S3-Workload serverseitige Verschlüsselung mit verwendet AWS KMS, finden Sie unter AWS KMS Limits im - AWS Key Management Service Entwicklerhandbuch Informationen zu den für Ihren Anwendungsfall unterstützten Anforderungsraten.

Die folgenden Themen beschreiben bewährte Verfahren und Designmuster zur Optimierung der Leistung von Anwendungen, die Amazon S3 verwenden. Die neuesten Informationen zur Leistungsoptimierung für Amazon S3 finden Sie unter Anleitungen zur Leistung von Amazon S3 und Leistungsdesignmuster für Amazon S3.

> **Note**
>
> Weitere Informationen zur Verwendung der Speicherklasse Amazon S3 Express One Zone mit Verzeichnis-Buckets finden Sie unter Was ist S3 Express One Zone? und Verzeichnis-Buckets.

Themen

Anleitungen zur Leistung von Amazon S3

Bei der Erstellung von Anwendungen, die Objekte zu Amazon S3 hochladen und davon abrufen, sollten Sie unsere bewährten Methoden befolgen, um die Leistung zu optimieren. Wir bieten auch detailliertere Leistungsdesignmuster.

Zur Erzielung der besten Leistung für Ihre Anwendung auf Amazon S3 empfehlen wir die folgenden Vorgehensweisen.

Themen

- Verwenden von Byte Range Fetches
- Wiederholungsanforderungen für latenzsensitive Anwendungen
- Kombinieren von Amazon S3 (Speicher) und Amazon EC2 (Datenverarbeitung) in derselben AWS-Region
- Verwenden von Amazon S3 Transfer Acceleration zur Minimierung der durch die Entfernung verursachten Latenz
- Verwendung der neuesten Version der AWS -SDKs

Messen der Leistung

Betrachten Sie für die Optimierung der Leistung den Netzwerkdurchsatz sowie die CPU- und DRAM-Anforderungen. Je nach der Mischung der Anforderungen für diese verschiedenen Ressourcen kann es sinnvoll sein, verschiedene Amazon EC2-Instance-Typen zu erwägen. Weitere Informationen zu verfügbaren Instance-Typen finden Sie unter Verfügbare Instance-Typen im Amazon EC2-Benutzerhandbuch für Linux-Instances.

Weiterhin ist es für die Messung der Leistung sinnvoll, die DNS-Lookup-Zeit, die Latenz und die Datenübertragungsgeschwindigkeit mithilfe von HTTP-Analysetools zu untersuchen.

Um die Leistungsanforderungen zu verstehen und die Leistung Ihrer Anwendung zu optimieren, können Sie auch die 503-Fehlerantworten überwachen, die Sie erhalten. Die Überwachung bestimmter Leistungsmetriken kann mit zusätzlichen Kosten verbunden sein. Weitere Informationen finden Sie unter Amazon S3 – Preise.

Überwachen der Anzahl der Antworten auf 503 (Slow Down)-Statusfehler

Um die Anzahl der Antworten auf 503-Statusfehler zu überwachen, die Sie erhalten, können Sie eine der folgenden Optionen verwenden:

- Verwenden Sie Amazon- CloudWatch Anforderungsmetriken für Amazon S3. Die CloudWatch Anforderungsmetriken enthalten eine Metrik für 5xx-Statusantworten. Weitere Informationen zu CloudWatch -Anforderungsmetriken finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.
- Verwenden Sie die Anzahl der 503 (Service Unavailable)-Fehler, die im Abschnitt „Erweiterte Metriken“ von Amazon S3 Storage Lens verfügbar ist. Weitere Informationen finden Sie unter Verwenden der Metriken von S3 Storage Lens zur Verbesserung der Leistung.

- Verwenden Sie die Amazon-S3-Serverzugriffsprotokollierung. Mit der Serverzugriffsprotokollierung können Sie alle Anforderungen filtern und überprüfen, die 503 (Internal Error)-Antworten erhalten. Sie können auch Amazon Athena verwenden, um Protokolle zu analysieren. Weitere Informationen zu Server-Zugriffsprotokollen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.

Durch die Überwachung der Anzahl der Statusfehlercodes HTTP 503 können Sie oft wertvolle Erkenntnisse darüber gewinnen, welche Präfixe, Schlüssel oder Buckets die meisten Drosselungsanforderungen erhalten.

Horizontale Skalierung von Speicherverbindungen

Die Verteilung von Anfragen über zahlreiche Verbindungen ist ein übliches Designmuster für die horizontale Skalierung der Leistung. Wenn Sie hochleistungsfähige Anwendungen erstellen, stellen Sie sich Amazon S3 als sehr großes verteiltes System vor, nicht als einen einzelnen Netzwerkendpunkt wie ein herkömmlicher Speicherserver. Sie erreichen die beste Leistung durch die Ausgabe mehrerer gleichzeitiger Anfragen an Amazon S3. Verteilen Sie diese Anfragen über separate Verbindungen, um die von Amazon S3 aus zugängliche Bandbreite zu maximieren. Amazon S3 hat keine Beschränkungen für die Anzahl der Verbindungen, die mit Ihrem Bucket hergestellt werden.

Verwenden von Byte Range Fetches

Mit dem `Range`-HTTP-Header in einer GET-Object-Anfrage können Sie einen Bytebereich von einem Objekt abrufen, wobei nur der angegebene Teil übertragen wird. Sie können gleichzeitige Verbindungen zu Amazon S3 verwenden, um verschiedene Bytebereiche aus demselben Objekt abzurufen. Dies hilft beim Erreichen eines höheren aggregierten Durchsatzes als bei einer einzelnen Anforderung eines ganzen Objekts. Der Abruf kleinerer Bereiche eines größeren Objekts ermöglicht Ihrer Anwendung auch die Verbesserung der Zeiten für Wiederholungsversuche, wenn Anforderungen unterbrochen werden. Weitere Informationen finden Sie unter Herunterladen von Objekten.

Typische Größen für Bytebereichanforderungen sind 8 oder 16 MB. Wenn für Objekte eine PUT-Aktion unter Verwendung eines mehrteiligen Uploads durchgeführt werden, ist es sinnvoll, die GET-Aktion mit denselben Teilgrößen (oder zumindest orientiert an den Teilgrenzen) durchzuführen, um eine optimale Leistung zu erzielen. GET-Anforderungen können sich direkt auf einzelne Teile richten, z. B., `GET ?partNumber=N`.

Wiederholungsanforderungen für latzenzsensitive Anwendungen

Aggressive Timeouts und Wiederholungsversuche sorgen für konsistente Latenz. Aufgrund des Umfangs von Amazon S3 gilt: Wenn die erste Anforderung langsam ist, nimmt eine alte Anfrage wahrscheinlich einen anderen Pfad und ist schnell erfolgreich. Die AWS SDKs verfügen über konfigurierbare Timeout- und Wiederholungswerte, die Sie an die Toleranzen Ihrer spezifischen Anwendung anpassen können.

Kombinieren von Amazon S3 (Speicher) und Amazon EC2 (Datenverarbeitung) in derselben AWS-Region

Obwohl Namen von S3-Buckets global eindeutig sind, wird jeder Bucket in einer Region gespeichert, die Sie bei dessen Erstellung auswählen. Um die Leistung zu optimieren, empfehlen wir Ihnen, AWS-Region nach Möglichkeit von Amazon EC2-Instances in derselben aus auf den Bucket zuzugreifen. Dies hilft bei der Reduzierung der Netzwerklatenz und der Datenübertragungskosten.

Weitere Informationen zu den Kosten von Datenübertragungen finden Sie unter Amazon S3 – Preise.

Verwenden von Amazon S3 Transfer Acceleration zur Minimierung der durch die Entfernung verursachten Latenz

Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration ermöglicht die schnelle, einfache und sichere Übertragung von Dateien über größere Entfernungen zwischen Ihrem Client und einem S3-Bucket. Transfer Acceleration nutzt die global verteilten Edge-Standorte in Amazon CloudFront. Sobald die Daten an einem Edge-Standort eingehen, werden sie über einen optimierten Netzwerkpfad an Amazon S3 weitergeleitet. Transfer Acceleration ist ideal für die regelmäßige Übertragung von Daten im Gigabyte- bis Terabyte-Bereich von Kontinent zu Kontinent geeignet. Die Funktion ist auch für Kunden nützlich, die Uploads in einen zentralen Bucket aus der ganzen Welt vornehmen.

Sie können das Amazon S3 Transfer Acceleration Speed Comparison Tool verwenden, um beschleunigte und nicht beschleunigte Upload-Geschwindigkeiten in allen Amazon S3-Regionen zu vergleichen. Das Speed Comparison-Tool verwendet mehrteilige Uploads, um eine Datei von Ihrem Browser in verschiedene Amazon S3-Regionen mit und ohne Amazon S3 Transfer Acceleration zu übertragen.

Verwendung der neuesten Version der AWS -SDKs

Die AWS SDKs bieten integrierte Unterstützung für viele der empfohlenen Richtlinien zur Optimierung der Amazon S3-Leistung. Die SDKs bieten eine einfachere API zur Nutzung von Amazon S3 aus einer Anwendung heraus und werden regelmäßig nach bewährten Verfahren aktualisiert. Beispielsweise enthalten die SDKs Logik für die automatische Wiederholung von Anforderungen bei HTTP 503-Fehlern und investieren in Code zur Reaktion auf langsame Verbindungen.

Die SDKs bieten dazu den Transfer Manager, der die horizontale Skalierung von Verbindungen automatisiert, um, wo möglich mithilfe von Bytebereichanforderungen, Tausende von Anfragen pro Sekunde zu erreichen. Es ist wichtig, die neueste Version der - AWS SDKs zu verwenden, um die neuesten Funktionen zur Leistungsoptimierung zu erhalten.

Sie können auch die Leistung optimieren, wenn Sie HTTP REST-API-Anforderungen verwenden. Bei der Verwendung der REST-API sollten Sie ebenfalls die bewährten Verfahren verwenden, die Teil der SDKs sind. Lassen Sie bei langsamen Anforderungen Timeouts und Wiederholungsversuche zu, sowie mehrere Verbindung, um den parallelen Abruf von Objektdaten zu ermöglichen. Informationen zur Verwendung der REST-API finden Sie in der Amazon Simple Storage Service API-Referenz.

Leistungsdesignmuster für Amazon S3

Beim Entwurf von Anwendungen zum Upload und Abruf von Objekten von Amazon S3 sollten Sie unsere bewährten Designmuster verwenden, um eine optimale Leistung für Ihre Anwendung zu erzielen. Dazu bieten wir Ihnen Leistungsanleitungen für Überlegungen zur Planung Ihrer Anwendungsarchitektur an.

Zur Optimierung der Leistung können Sie die folgenden Designmuster verwenden.

Themen

- Verwendung von Caching für Inhalte mit häufigen Zugriffen
- Timeouts und Wiederholungsversuche für latenzsensitive Anwendungen
- Horizontale Skalierung und Anforderungsparallelisierung für hohen Durchsatz
- Verwendung von Amazon S3 Transfer Acceleration zur Beschleunigung geographisch disparater Datenübertragungen

Verwendung von Caching für Inhalte mit häufigen Zugriffen

Viele Anwendungen, die Daten in Amazon S3 speichern, stellen einen „Arbeitssatz“ der Daten bereit, der von Benutzern häufig angefragt wird. Wenn ein Workload wiederholte GET-Anforderungen für einen gemeinsamen Satz von Objekten sendet, können Sie einen Cache wie Amazon CloudFront, Amazon ElastiCache oder verwenden, AWS Elemental MediaStore um die Leistung zu optimieren. Die erfolgreiche Cache-Nutzung kann zu niedriger Latenz und zu hohen Datenübertragungsraten führen. Anwendungen, die die Zwischenspeicherung verwenden, senden auch weniger direkte Anforderungen an Amazon S3, was zur Senkung der Anfragekosten beitragen kann.

Amazon CloudFront ist ein schnelles Content Delivery Network (CDN), das Daten aus Amazon S3 in einer großen Menge geografisch verteilter Points of Presence () transparent zwischenspeichertPoPs. Wenn auf Objekte aus mehreren Regionen oder über das Internet zugegriffen werden kann, CloudFront erlaubt , dass Daten in der Nähe der Benutzer zwischengespeichert werden, die auf die Objekte zugreifen. Dies kann zu hoher Leistung bei der Bereitstellung beliebter Amazon S3-Inhalte führen. Weitere Informationen zu CloudFrontfinden Sie im Amazon- CloudFront Entwicklerhandbuch.

Amazon ElastiCache ist ein verwalteter In-Memory-Cache. Mit können ElastiCacheSie Amazon EC2-Instances bereitstellen, die Objekte im Speicher zwischenspeichern. Dieses Caching führt zur Reduzierung der GET-Latenz im Bereich mehrerer Größenordnungen und zu einer erheblichen Zunahme des Downloaddurchsatzes. Um zu verwenden ElastiCache, ändern Sie die Anwendungslogik, um sowohl den Cache mit aktiven Objekten zu füllen als auch den Cache auf aktive Objekte zu überprüfen, bevor Sie sie von Amazon S3 anfordern. Beispiele für die Verwendung von ElastiCache zur Verbesserung der Leistung von Amazon S3 GET finden Sie im Blogbeitrag Turbo Charge Amazon S3 with Amazon ElastiCache for Redis .

AWS Elemental MediaStore ist ein Zwischenspeicherungs- und Inhaltsverteilungssystem, das speziell für Videoworkflows und die Medienbereitstellung von Amazon S3 entwickelt wurde. MediaStore stellt end-to-end Speicher-APIs speziell für Videos bereit und wird für leistungsempfindliche Video-Workloads empfohlen. Weitere Informationen zu MediaStorefinden Sie im AWS Elemental MediaStore -Benutzerhandbuch.

Timeouts und Wiederholungsversuche für latenzsensitive Anwendungen

Es gibt bestimmte Situationen, in denen eine Anwendung eine Antwort von Amazon S3 erhält, die darauf hinweist, dass ein Wiederholungsversuch erforderlich ist. Amazon S3 ordnet Bucket- und Objektnamen den damit verbundenen Objektdaten zu. Wenn eine Anwendung hohe Anforderungsraten generiert (typischerweise dauerhaft über 5.000 Anforderungen pro Sekunde für

eine kleine Zahl von Objekten), erhält sie möglicherweise HTTP 503 Slowdown-Antworten. Wenn solche Fehler auftreten, implementiert jedes AWS -SDK eine automatische Wiederholungsversuch-Logik mit exponentiellem Backoff. Wenn Sie kein AWS -SDK verwenden, sollten Sie eine Wiederholungsversuch-Logik implementieren, wenn Sie den HTTP-Fehler 503 erhalten. Weitere Informationen zu Back-off-Techniken finden Sie unter Wiederholversuche bei Fehlern und Exponentielles Backoff in AWS im Allgemeine Amazon Web Services-Referenz.

Amazon S3 wird automatisch in Reaktion auf andauernde neue Anforderungsraten skaliert und optimiert so die Leistung in dynamischer Weise. Während Amazon S3 Optimierungen für eine neue Anforderungsrate durchführt, erhalten Sie temporär HTTP 503-Anforderungsantworten, bis die Optimierung abgeschlossen ist. Nachdem Amazon S3 die Leistung intern für die neue Anfragerate optimiert hat, werden alle Anfragen generell ohne Wiederholungsversuche bereitgestellt.

Für latenzsensitive Anwendungen empfiehlt Amazon S3 Nachverfolgung und aggressive Wiederholungsversuche bei langsameren Vorgängen. Wenn Sie eine Anfrage wiederholen, empfehlen wir die Verwendung einer neuen Verbindung mit Amazon S3 und die Durchführung eines neuen DNS-Lookup-Vorgangs.

Wenn Sie sehr große Anforderungen unterschiedlicher Größe (beispielsweise mit mehr als 128 MB) durchführen, sollten Sie den erreichten Durchsatz nachverfolgen und für die langsamsten 5 Prozent der Anforderungen Wiederholungsversuche durchführen. Wenn Sie kleinere Anforderungen (etwa unter 512 KB) durchführen, bei denen die mittleren Latenzen oft im zweistelligen Millisekundenbereich liegen, ist es sinnvoll, nach zwei Sekunden eine GET- oder PUT-Operation zu wiederholen. Wenn weitere Wiederholungsversuche erforderlich sind, ist ein Backoff die beste Lösung. So empfehlen wir beispielsweise die Ausgabe eines Wiederholungsversuchs nach zwei Sekunden und eines zweiten Versuchs nach weiteren vier Sekunden.

Wenn Ihre Anwendung Anfragen mit fester Größe an Amazon S3 sendet, sollten Sie konsistentere Reaktionszeiten für diese einzelnen Anfragen erwarten. In diesem Fall ist es eine einfache Strategie, das langsamste Prozent der Anforderungen zu identifizieren und diese zu wiederholen. Selbst ein einziger Wiederholungsversuch ist oft erfolgreich für die Reduzierung der Latenz.

Wenn Sie AWS Key Management Service (AWS KMS) für die serverseitige Verschlüsselung verwenden, finden Sie unter Limits im -AWS Key Management Service Entwicklerhandbuch Informationen zu den Anforderungsraten, die für Ihren Anwendungsfall unterstützt werden.

Horizontale Skalierung und Anforderungsparallelisierung für hohen Durchsatz

Amazon S3 ist ein sehr großes verteiltes System. Um diese Größe zu nutzen, sollten Sie parallele Anforderungen horizontal zu den Amazon S3-Service-Endpunkten skalieren. Zusätzlich zur Verteilung der Anforderungen in Amazon S3 hilft dieses Skalierungskonzept dabei, die Last über mehrere Pfade im Netzwerk zu verteilen.

Für Übertragungen mit hohem Durchsatz empfiehlt Amazon S3 die Verwendung von Anwendungen mit mehreren Verbindungen für die parallele Ausführung von GET- und PUT-Aktionen. Dies wird beispielsweise von Amazon S3 Transfer Manager im AWS Java SDK unterstützt, und die meisten anderen AWS SDKs bieten ähnliche Konstrukte. Für manche Anwendungen können Sie parallele Verbindungen dadurch erreichen, dass Sie Anforderungen gleichzeitig in verschiedenen Anwendungsthreads oder in verschiedenen Anwendungsinstances starten. Das beste Konzept hängt von Ihrer Anwendung und der Struktur der Objekte ab, auf die Sie zugreifen.

Sie können die - AWS SDKs verwenden, um GET- und PUT-Anfragen direkt auszugeben, anstatt die Verwaltung von Übertragungen im AWS SDK zu nutzen. Dieses Konzept ermöglicht die direktere Abstimmung Ihrer Workloads, bei gleichzeitiger Nutzung des SDK-Supports für Wiederholungsversuche und den Umgang mit eventuell auftretenden HTTP 503-Antworten. Generell gilt: Wenn Sie in einer Region große Objekte von Amazon S3 zu Amazon EC2 herunterladen, sollten Sie gleichzeitige Anfragen für Bytebereiche eines Objekts in der Größe von 8-16 MB starten. Starten Sie eine gleichzeitige Anforderung für jeweils 85-90 MB/s des gewünschten Netzwerkdurchsatzes. Zur Ausnutzung einer Netzwerkschnittstellenkarte (NIC) mit 10 Gb/s können Sie etwa 15 gleichzeitige Anforderungen über separate Verbindungen durchführen. Sie können die gleichzeitigen Anforderungen über mehr Verbindungen hochfahren, um schnellere NICs zu nutzen, etwa 25 oder 100 Gb/s.

Die Messung der Leistung ist wichtig für die Einstellung der Anzahl der gleichzeitig auszugebenden Anforderungen. Wir empfehlen, mit jeweils einer einzigen Anforderung zu beginnen. Messen Sie die erreichte Netzwerkbandbreite und die Nutzung weiterer Ressourcen durch Ihre Anwendung bei der Verarbeitung der Daten. Sie können dann die Engpassressource (die Ressource mit der höchsten Nutzung) identifizieren und so die Anzahl der Anforderungen ermitteln, die wahrscheinlich nützlich waren. z. B.: Wenn die Verarbeitung einzelner Anforderungen zu einer CPU-Nutzung von 25 Prozent führt, zeigt dies an, dass bis zu vier gleichzeitige Anforderungen möglich sind. Messungen sind sehr wichtig, und es lohnt sich, die Ressourcennutzung zu ermitteln, wenn die Anforderungsrate erhöht wird.

Wenn ihre Anwendung Anforderungen direkt an Amazon S3 mit der REST-API ausgibt, sollten Sie einen Pool von HTTP-Verbindungen verwenden und jede Verbindung für eine Serie von Anforderungen wiederverwenden. Die Vermeidung der Einrichtung von Verbindungen für jede Anforderung macht TCP-Slow-Starts und Secure Sockets Layer (SSL)-Handshakes bei jeder Anforderung überflüssig. Informationen zur Verwendung der REST-API finden Sie in der Amazon Simple Storage Service API-Referenz.

Schließlich ist es auch sinnvoll, auf den DNS zu achten und genau zu prüfen, ob Anfragen über einen großen Pool von Amazon S3-IP-Adressen verteilt werden. DNS-Anforderungen für Amazon S3 durchlaufen eine lange Liste von IP-Endpunkten. Das Caching von Resolvern oder Anwendungscode, der eine einzelne IP-Adresse wiederverwendet, profitiert nicht von der Adressendiversität und dem daraus resultierenden Lastenausgleich. Network Utility-Tools wie das `netstat`-Befehlszeilentool können die IP-Adressen anzeigen, die für die Kommunikation mit Amazon S3 verwendet werden, und wir bieten Anleitungen zu den zu verwendenden DNS-Konfigurationen. Weitere Informationen zu diesen Anleitungen finden Sie unter Senden von Anforderungen.

Verwendung von Amazon S3 Transfer Acceleration zur Beschleunigung geographisch disparater Datenübertragungen

Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration ist effektiv bei der Minimierung oder Beseitigung der durch geographische Entfernung zwischen global verteilten Clients und einer regionalen Anwendung mit Amazon S3 verursachten Latenz. Transfer Acceleration verwendet die global verteilten Edge-Standorte in CloudFront für den Datentransport. Das AWS Edge-Netzwerk verfügt über Präsenzpunkte an mehr als 50 Standorten. Heute wird es verwendet, um Inhalte über zu verteilen CloudFront und schnelle Antworten auf DNS-Abfragen an Amazon Route 53 bereitzustellen.

Dazu hilft das Edge-Netzwerk bei der Beschleunigung von Datenübertragungen zu und aus Amazon S3. Es ist ideal für Anwendungen, die Daten über oder zwischen Kontinenten übertragen, eine schnelle Internetverbindung nutzen, große Objekte verwenden oder zahlreiche Inhalte hochladen müssen. Sobald die Daten an einem Edge-Standort eingehen, werden sie über einen optimierten Netzwerkpfad an Ihren Amazon S3-Bucket weitergeleitet. Allgemein gilt: Je weiter Sie von einer Amazon S3-Region entfernt sind, um so größer ist die Geschwindigkeits-Verbesserung, die Sie von Transfer Acceleration erwarten können.

Sie können Transfer Acceleration auf neuen oder bestehenden Buckets einrichten. Sie können einen separaten Amazon S3 Transfer Acceleration-Endpunkt verwenden, um die AWS Edge-Standorte zu

verwenden. Die beste Möglichkeit zur Prüfung, ob Transfer Acceleration die Client-Anfrageleistung erhöht, ist die Verwendung des Amazon S3-Transfer-Acceleration-Speed-Comparison-Tools. Netzwerkkonfigurationen und Bedingungen variieren von zeit zu Zeit und von Standort zu Standort. Sie werden daher nur für Übertragungen belastet, bei denen Amazon S3 Transfer Acceleration Ihre Upload-Leistung potentiell verbessern kann. Informationen zur Verwendung von Transfer Acceleration mit verschiedenen AWS SDKs finden Sie unter Aktivieren und Verwenden von S3 Transfer Acceleration.

Was ist Amazon S3 on Outposts?

AWS Outposts ist ein vollständig verwalteter Service, der die gleiche AWS-Infrastruktur, AWS-Services, APIs und Tools für praktisch jedes Rechenzentrum, jeden Co-Location-Raum oder jede On-Premises-Einrichtung für ein wirklich konsistentes Hybrid-Erlebnis bietet .AWS Outposts ist ideal für Workloads, die Zugriff auf lokale Systeme mit geringer Latenz, lokale Datenverarbeitung, Datenresidenz und Migration von Anwendungen mit lokalen Systemabhängigkeiten erfordern. Weitere Informationen finden Sie unter Was ist AWS Outposts? im AWS Outposts-Benutzerhandbuch.

Mit Amazon S3 on Outposts können Sie S3-Buckets in Ihren Outposts erstellen und Objekte einfach On-Premises speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens `OUTPOSTS`. Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihren Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outposts-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC).

Sie können bei Outposts-Buckets dieselben APIs und Funktionen wie in Amazon S3 verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden.

Funktionsweise von S3 on Outposts

S3 on Outposts ist ein Objektspeicherdienst, der Daten als Objekte in Buckets in Ihrem Outpost speichert. Ein Objekt ist eine Datendatei und alle Metadaten, die diese Datei beschreiben. Ein Bucket ist ein Container für Objekte.

Um Ihre Daten in S3 on Outposts zu speichern, müssen Sie zunächst einen Bucket erstellen. Beim Erstellen des Buckets geben Sie einen Bucket-Namen und den Outpost an, der den Bucket enthält. Um auf Ihren S3-on-Outposts-Bucket zuzugreifen und Objektoperationen durchzuführen, erstellen und konfigurieren Sie als Nächstes einen Zugriffspunkt. Sie müssen auch einen Endpunkt erstellen, um Anfragen an Ihren Zugriffspunkt weiterzuleiten.

Zugriffspunkte vereinfachen den Datenzugriff für jeden AWS-Service oder Kundenanwendung, die Daten in S3 speichert. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind und mit denen Sie Objektvorgänge ausführen können, z. B. `GetObject` und `PutObject`. Jeder Zugriffspunkt verfügt über unterschiedliche Berechtigungen und Netzwerkkontrollen.

Sie können Ihre S3-on-Outposts-Buckets, Zugriffspunkte und Endpunkte erstellen und verwalten, indem Sie die AWS Management Console, AWS CLI, AWS SDKs oder REST API verwenden. Um Objekte in Ihren S3-on-Outposts-Bucket hochzuladen und zu verwalten, können Sie die AWS CLI, AWS SDKs oder REST API verwenden.

Regionen

Während der AWS Outposts-Bereitstellung erstellen Sie oder AWS eine Service-Link-Verbindung, die Ihren Outpost wieder mit dem von Ihnen gewählten AWS-Region oder der Outposts-Heimatregion für Bucket-Operationen und Telemetrie verbindet. Ein Outpost ist auf Konnektivität zum übergeordneten AWS-Region angewiesen. Das Outposts-Rack ist nicht für getrennte Operationen oder Umgebungen mit eingeschränkter oder keiner Konnektivität ausgelegt. Weitere Informationen finden Sie unter Outpost-Konnektivität zu AWS-Regionen im AWS Outposts Benutzerhandbuch.

Buckets

Ein Bucket ist ein Behälter für Objekte, die in S3 on Outposts gespeichert werden. Sie können beliebig viele Objekte in einem Bucket speichern und bis zu 100 Buckets pro Konto in einem Outpost haben.

Wenn Sie einen Bucket erstellen, geben Sie einen Bucket-Namen ein und wählen den Outpost, in dem der Bucket angelegt werden soll. Der Name eines erstellten Buckets oder sein Outpost kann nicht nachträglich geändert werden. Bucket-Namen müssen den Regeln für die Benennung von Amazon-S3-Buckets folgen. In S3 on Outposts sind die Bucket-Namen für einen Outpost und AWS-Konto einmalig. `outpost-id`, `account-id` und der Bucket-Name müssen die S3-on-Outposts-Buckets identifizieren.

Im folgenden Beispiel wird das Format des Amazon-Ressourcennamens (ARN) für S3-on-Outposts-Buckets gezeigt. Der ARN besteht aus der Region, in der sich Ihr Outpost befindet, Ihrem Outpost-Konto, der Outpost-ID und dem Bucket-Namen.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Zugriffspunkt-ARN oder den Zugriffspunktalias. Weitere Informationen über Zugriffspunkt-Aliasse finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das Format des Zugriffspunkt-ARN für S3 on Outposts, das die `outpost-id`, die `account-id` und den Namen des Zugriffspunkts umfasst:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen über Buckets finden Sie unter Arbeiten mit S3-on-Outposts-Buckets.

Objekte

Objekte sind die Grundeinheiten, die in S3 on Outposts gespeichert sind. Objekte bestehen aus Objekt- und Metadaten. Metadaten bestehen aus mehreren Name/Wert-Paaren, die das Objekt beschreiben. Dazu gehören Standardmetadaten wie das Datum der letzten Aktualisierung und HTTP-Standardmetadaten wie `Content-Type`. Sie können bei der Speicherung des Objekts auch benutzerdefinierte Metadaten angeben. Ein Objekt wird innerhalb eines Buckets eindeutig durch einen Schlüssel (oder Namen) identifiziert.

Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Schlüssel

Ein Objektschlüssel (oder Schlüsselname) ist der eindeutige Bezeichner für ein Objekt in einem Bucket. Jedes Objekt in einem Bucket besitzt genau einen Schlüssel. Jedes Objekt wird durch die Kombination aus Bucket und Objektschlüssel eindeutig identifiziert.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Objekte, das den AWS-Region-Code für die Region, in der sich der Outpost befindet, AWS-Konto-ID, Outpost-ID, Bucket-Name und Objektschlüssel beinhaltet:

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/bucket/DOC-EXAMPLE-BUCKET1/object/myobject
```

Weitere Informationen über Objektschlüssel finden Sie unter Arbeiten mit S3-on-Outposts-Objekten.

S3-Versioning

Sie können die S3-Versionsverwaltung für Outposts-Buckets verwenden, um mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Mit S3-Versioning können Sie jede Version jedes in Ihren Buckets gespeicherten Objekts beibehalten, abrufen und wiederherstellen. Mit der S3-Versionsverwaltung können Sie Versionen sowohl nach unbeabsichtigten Benutzeraktionen als auch nach Anwendungsfehlern problemlos wiederherstellen.

Weitere Informationen finden Sie unter Verwalten der S3-Versionierung für Ihren S3-on-Outposts-Bucket.

Versions-ID

Wenn Sie die S3-Versionsverwaltung in einem Bucket aktivieren, generiert S3 on Outposts eine eindeutige Versions-ID für jedes Objekt, das dem Bucket hinzugefügt wird. Objekte, die zum Zeitpunkt der Aktivierung des Versioning bereits im Bucket vorhanden waren, haben die Versions-ID`null` . Wenn Sie diese (oder andere) Objekte mit anderen Operationen wie PutObject verändern, erhalten die neuen Objekte eine eindeutige Versions-ID.

Weitere Informationen finden Sie unter Verwalten der S3-Versionierung für Ihren S3-on-Outposts-Bucket.

Speicherklasse und Verschlüsselung

S3 in Outposts bietet eine neue Speicherklasse: S3 Outposts (`OUTPOSTS`). Die Speicherklasse S3 Outposts ist nur für Objekte verfügbar, die in Buckets auf AWS Outposts gespeichert sind. Wenn Sie versuchen, andere S3-Speicherklassen mit S3 on Outposts zu verwenden, gibt S3 on Outposts den Fehler `InvalidStorageClass` aus.

Objekte, die in der Speicherklasse S3 Outposts (`OUTPOSTS`) gespeichert sind, werden standardmäßig mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) verschlüsselt. Weitere Informationen finden Sie unter Datenverschlüsselung in S3 on Outposts.

Bucket-Richtlinie

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende AWS Identity and Access Management-(IAM)-Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt.

Bucket-Richtlinien verwenden JSON-basierte IAM-Richtliniensprache, die standardmäßig in AWS ist. Sie können Bucket-Richtlinien verwenden, um Berechtigungen für die Objekte in einem Bucket hinzuzufügen oder zu verweigern. Bucket-Richtlinien erlauben oder verweigern Anforderungen basierend auf den Elementen in der Richtlinie. Diese Elemente können den Anforderer, S3-on-Outposts-Aktionen, Ressourcen und Aspekte oder Bedingungen der Anforderung beinhalten (z. B. die IP-Adresse, die für die Anforderung verwendet wird). Sie können beispielsweise eine Bucket-Richtlinie erstellen, die kontoübergreifende Berechtigungen zum Hochladen von Objekten in einen S3-on-Outposts-Bucket gewährt, während gleichzeitig sichergestellt wird, dass der Bucket-Eigentümer die volle Kontrolle über die hochgeladenen Objekte hat. Weitere Informationen finden Sie unter Beispiele für Bucket-Richtlinien.

In Ihrer Bucket-Richtlinie können Sie Platzhalterzeichen (`*`) für ARNs und andere Werte verwenden, um Berechtigungen für eine Teilmenge von Objekten zu erteilen. Sie können beispielsweise den Zugriff auf Gruppen von Objekten steuern, die mit einem gemeinsamen Präfixbeginnen oder mit einer bestimmten Erweiterung wie `.html` enden.

S3-on-Outposts-Zugriffspunkte

S3-on-Outposts-Zugriffspunkte sind benannte Netzwerkendpunkte mit dedizierten Zugriffsrichtlinien, die beschreiben, wie mit diesem Endpunkt auf Daten zugegriffen werden kann. Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in S3 on Outposts. Zugriffspunkte sind Buckets zugeordnet, mit denen Sie S3-Objektvorgänge ausführen können, z. B. `GetObject` und `PutObject`.

Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Zugriffspunkt-ARN oder den Zugriffspunktalias. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Jeder Zugriffspunkt verfügt über unterschiedliche Berechtigungen und Netzwerkkontrollen, die S3 on Outposts auf alle Anforderungen anwendet, die über diesen Zugriffspunkt eingehen. Jeder Zugriffspunkt erzwingt eine benutzerdefinierte Zugriffspunktrichtlinie, die in Verbindung mit der Bucket-Richtlinie funktioniert, die dem zugrunde liegenden Bucket zugeordnet ist.

Weitere Informationen finden Sie unter Zugriff auf Ihre S3-on-Outposts-Buckets und -Objekte.

Funktionen von S3 on Outposts

Zugriffsverwaltung

S3 on Outposts bietet Funktionen für die Überwachung und Verwaltung des Zugriffs auf Ihre Buckets und Objekte. Standardmäßig werden S3-on-Outposts-Buckets und -Objekte als privat eingestuft. Sie haben nur Zugriff auf die S3-on-Outposts-Ressourcen, die Sie erstellen.

Um detaillierte Ressourcenberechtigungen zu erteilen, die Ihren speziellen Anwendungsfall unterstützen, oder um die Berechtigungen Ihrer S3-on-Outposts-Ressourcen zu überprüfen, können Sie die folgenden Funktionen verwenden.

- S3 öffentlichen Zugriff blockieren – Blockieren Sie den öffentlichen Zugriff auf Buckets und Objekte. Für Buckets auf Outposts ist „Öffentlichen Zugriff blockieren“ standardmäßig aktiviert.
- AWS Identity and Access Management(IAM) – IAM ist ein Webservice, der Ihnen hilft, den Zugriff auf AWS-Ressourcen zu steuern, einschließlich S3-on-Outposts-Ressourcen. Mit IAM können Sie Berechtigungen, die festlegen, auf welche AWS-Ressourcen Benutzer zugreifen dürfen, zentral verwalten. Sie verwenden IAM, um zu steuern, wer authentifiziert (angemeldet) und autorisiert (Berechtigungen besitzt) ist, Ressourcen zu nutzen.

- S3-on-Outposts-Zugriffspunkte – Verwalten Sie den Datenzugriff auf freigegebene Datensätze in S3 on Outposts. Zugriffspunkte sind benannte Netzwerkendpunkte mit dedizierten Zugriffsrichtlinien. Zugriffspunkte sind Buckets zugeordnet und können für Objektvorgänge verwendet werden, z. B. `GetObject` und `PutObject`.
- Bucket-Richtlinien – Verwenden Sie die IAM-basierte Richtliniensprache, um ressourcenbasierte Berechtigungen für Ihre S3-Buckets und die darin enthaltenen Objekte zu konfigurieren.
- AWS Resource Access Manager(AWS RAM) – Geben Sie Ihre Kapazität von S3 on Outposts innerhalb von AWS-Konten, Ihrer Organisation oder Organisationseinheiten (OUs) in AWS Organizations sicher frei.

Speicherprotokollierung und Überwachung

S3 on Outposts bietet Protokollierungs- und Überwachungstools, mit denen Sie überwachen und steuern können, wie Ihre S3-on-Outposts-Ressourcen verwendet werden. Weitere Informationen finden Sie unter Überwachungstools.

- Amazon-CloudWatch-Metriken für S3 on Outposts – Verfolgen Sie den Betriebszustand Ihrer Ressourcen und zeigen Sie die verfügbaren Kapazitäten an.
- Amazon CloudWatch Events für S3 on Outposts – Erstellen Sie für jedes API-Ereignis von S3 on Outposts eine Regel, um Benachrichtigungen über alle unterstützten Ziele von CloudWatch Events zu erhalten, einschließlich Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS) und AWS Lambda.
- AWS CloudTrail-Protokolle für S3 on Outposts – Zeichnen Sie Aktionen eines Benutzers, einer Rolle oder eines AWS-Service in S3 on Outposts auf. CloudTrail-Protokolle bieten Ihnen detailliertes API-Tracking für Vorgänge auf S3-Bucket- und -Objektebene.

Starke Konsistenz

Amazon S3 bietet eine hohe Lesen-nach-Schreiben-Konsistenz für PUT- und DELETE-Anforderungen von Objekten in Ihrem S3-on-Outposts-Bucket in allen AWS-Regionen. Dieses Verhalten gilt sowohl für Schreibvorgänge neuer Objekte als auch für PUT-Anforderungen, die vorhandene Objekte überschreiben, und DELETE-Anforderungen. Darüber hinaus sind S3-on-Outposts-Objektmarkierungen und Objekt-Metadaten (z. B. das HEAD-Objekt) sehr konsistent. Weitere Informationen finden Sie unter Amazon S3-Datenkonsistenzmodell.

Zugehörige Services

Nachdem Sie Daten in S3 on Outposts hochgeladen haben, können Sie sie mit anderen AWS-Services nutzen. Häufig genutzte Services:

- Amazon Elastic Compute Cloud (Amazon EC2) – Bietet sichere und skalierbare Rechenkapazität in der AWS Cloud. Amazon EC2 reduziert die Notwendigkeit, im Voraus in Hardware investieren zu müssen. Daher können Sie Anwendungen schneller entwickeln und bereitstellen. Mit Amazon EC2 können Sie so viele oder so wenige virtuelle Server starten, wie Sie benötigen, die Sicherheit und das Netzwerk konfigurieren und den Speicher verwalten.
- Amazon Elastic Block Store (Amazon EBS) on Outposts – Verwenden Sie Amazon EBS local snapshots on Outposts, um Snapshots von Volumes auf einem Outpost lokal in S3 on Outpost zu speichern.
- Amazon Relational Database Service (Amazon RDS) on Outposts – Verwenden Sie lokale Amazon RDS-Backups, um Ihre Amazon RDS-Backups lokal in Ihrem Outpost zu speichern.
- AWS DataSync – Automatisieren Sie die Übertragung von Daten zwischen Ihren Outposts und AWS-Regionen. Dabei können Sie auswählen, was übertragen werden soll, wann es übertragen werden soll und wie viel Netzwerkbandbreite verwendet werden soll. S3 on Outposts ist in AWS DataSync integriert. Für On-Premises-Anwendungen, die eine lokale Verarbeitung mit hohem Durchsatz erfordern, bietet S3 on Outposts On-Premises-Objektspeicher, um Datenübertragungen zu minimieren und einen Puffer gegen Netzwerkschwankungen zu bieten, und ermöglicht Ihnen gleichzeitig, Daten einfach zwischen Outposts und AWS-Regionen zu übertragen.

Zugriff auf S3 on Outposts

Sie können mit S3 on Outposts auf eine der folgenden Arten arbeiten:

AWS Management Console

Die Konsole ist eine webbasierte Benutzeroberfläche für die Verwaltung von S3 on Outposts und AWS-Ressourcen. Wenn Sie sich für ein AWS-Konto registriert haben, können Sie auf S3 on Outposts zugreifen, indem Sie sich bei AWS Management Console anmelden und auf der Startseite der AWS Management Console S3 auswählen. Wählen Sie dann Outposts buckets (Outposts-Buckets) aus dem linken Navigationsbereich aus.

AWS Command Line Interface

Sie können die Befehlszeilen-Tools von AWS verwenden, um Befehle in der Befehlszeile Ihres Systems auszugeben, mit denen AWS-Aufgaben (einschließlich S3) durchgeführt werden.

Das AWS Command Line Interface (AWS CLI) stellt Befehle für zahlreiche AWS-Services bereit. Die AWS CLI wird unter Windows, macOS und Linux unterstützt. Informationen zu den ersten Schritten finden Sie im AWS Command Line Interface-Benutzerhandbuch. Weitere Informationen zu den Befehlen, die Sie mit S3 on Outposts verwenden können, finden Sie unter s3api, s3control und s3outposts in der AWS CLI-Befehlsreferenz.

AWS-SDKs

AWS stellt SDKs (Software Development Kits) zur Verfügung, die aus Bibliotheken und Beispiel-Codes für verschiedene Programmiersprachen und Plattformen (Java, Python, Ruby, .NET, iOS, Android usw.) bestehen. Die AWS-SDKs eignen sich hervorragend zur Einrichtung des programmgesteuerten Zugriffs auf S3 on Outposts und AWS. Da S3 on Outposts dieselben SDKs wie Amazon S3 verwendet, bietet S3 on Outposts dieselben S3-APIs, Automatisierung und Tools und somit eine einheitliche Erfahrung.

S3 on Outposts ist ein REST-Service. Sie können Anfragen an S3 on Outposts über die AWS-SDK-Bibliotheken senden, die die zugrunde liegende Amazon REST-API umschließen, und somit Ihre Programmieraufgaben vereinfachen. Beispielsweise übernehmen die SDKs Aufgaben wie das Berechnen von Signaturen, das kryptografische Signieren von Anforderungen, das Verwalten von Fehlern und das automatische erneute Ausführen von Anforderungen. Weitere Informationen über die AWS-SDKs, das Herunterladen und die Installation finden Sie unter Tools zur Erstellung von AWS.

Bezahlung für S3 on Outposts

Sie können viele verschiedene AWS Outposts-Rack-Konfigurationen mit einer Kombination von Amazon-EC2-Instance-Typen, universellen Amazon-EBS-SSD-Laufwerk-Volumes (Solid State Drive) (gp2) und S3 on Outposts erwerben. Die Preise beinhalten Lieferung, Installation, Wartung von Infrastruktur-Services sowie Softwarepatches und Upgrades.

Weitere Informationen finden Sie in der Preisliste für AWS Outposts-Racks.

Nächste Schritte

Weitere Informationen zur Arbeit mit S3 on Outposts finden Sie in den folgenden Themen:

Einrichten Ihres Outposts

Um mit Amazon S3 on Outposts zu beginnen, benötigen Sie einen Outpost mit Amazon-S3-Kapazität, der in Ihrer Einrichtung bereitgestellt wird. Weitere Informationen zu den Optionen für die Bestellung eines Outposts und von S3-Kapazitäten finden Sie unter AWS Outposts. Zum Überprüfen, ob Ihre Outposts über S3-Kapazität verfügen, können Sie den API-Aufruf ListOutpostsWithS3 verwenden. Technische Daten und weitere Informationen dazu, wie sich S3 on Outposts von Amazon S3 unterscheidet, finden Sie unter Wie unterscheidet sich Amazon S3 on Outposts von Amazon S3?.

Weitere Informationen finden Sie unter den folgenden Themen.

Themen

Einen neuen -Outpost bestellen

Wenn Sie einen neuen Outpost mit S3-Kapazität bestellen müssen, lesen Sie Preisliste für AWS Outposts-Racks, um die Kapazitätsoptionen für Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Block Store (Amazon EBS) und Amazon S3 zu verstehen.

Nach der Auswahl der Konfiguration führen Sie die Schritte unter Erstellen eines Outposts und Bestellen von Outpost-Kapazitäten im AWS Outposts-Benutzerhandbuch aus.

Wie unterscheidet sich Amazon S3 on Outposts von Amazon S3?

Amazon S3 on Outposts stellt Ihrer lokalen AWS Outposts-Umgebung Objektspeicher bereit. Mit S3 on Outposts können Sie die Anforderungen im Hinblick auf lokale Verarbeitung, Datenaufbewahrung und anspruchsvolle Leistung erfüllen, indem Daten in der Nähe von lokalen Anwendungen bleiben. Mithilfe von Amazon-S3-APIs und -Funktionen erleichtert S3 on Outposts das Speichern, Sichern, Markieren, Melden sowie die Kontrolle des Zugriffs auf die Daten in Ihren Outposts. Außerdem wird die AWS-Infrastruktur für Ihre On-Premises-Einrichtung zugunsten eines konsistenten Hybrid-Erlebnisses erweitert.

Weitere Informationen zu den Alleinstellungsmerkmalen von S3 on Outposts finden Sie in den folgenden Themen.

Themen

Spezifikationen für S3 auf Outposts

- Die maximale Outpost-Bucket-Größe beträgt 50 TB.
- Die maximale Anzahl vin Outpost-Buckets beträgt 100 pro AWS-Konto.
- Auf Outpost-Buckets kann nur über Zugriffs- und Endpunkte zugegriffen werden.
- Die maximale Anzahl von Zugriffspunkten pro Outpost-Bucket beträgt 10.
- Zugriffspunkt-Richtlinien sind auf eine Größe von 20 KB beschränkt.
- Der Outpost-Eigentümer kann den Zugriff innerhalb Ihrer Organisation in AWS Organizations mithilfe von AWS Resource Access Manager verwalten. Alle Konten, die Zugriff auf den Außenposten benötigen, müssen sich innerhalb derselben Organisation befinden wie das Eigentümerkonto in AWS Organizations.
- Das S3 in Outposts-Bucket-Eigentümerkonto ist immer der Eigentümer aller Objekte im Bucket.
- Nur das S3 in Outposts-Bucket-Eigentümerkonto kann Vorgänge für den Bucket ausführen.
- Die Objektgrößenbegrenzungen entsprechen denen von Amazon S3.

- Alle auf S3 auf Outposts gespeicherten Objekte werden in der Speicherklasse `OUTPOSTS` gespeichert.
- Standardmäßig werden alle in der Speicherklasse `OUTPOSTS` gespeicherten Objekte mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) gespeichert. Sie können auch explizit auswählen, Objekte mithilfe serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) zu speichern.
- Wenn nicht genügend Speicherplatz vorhanden ist, um ein Objekt in Ihrem Outpost zu speichern, gibt die API eine Ausnahme wegen unzureichender Kapazität (ICE) zurück.

Von S3 unterstützte API-Operationen auf Outposts

Eine Liste der von S3 on Outposts unterstützten API-Operationen finden Sie unter API-Vorgänge in Amazon S3 on Outposts.

Amazon-S3-Funktionen, die von S3 auf Outposts nicht unterstützt werden

Mehrere Amazon-S3-Funktionen werden derzeit von Amazon S3 auf Outposts nicht unterstützt. Versuche, sie zu verwenden, werden zurückgewiesen.

- Zugriffskontrolllisten (ACLs)
- Cross-Origin Resource Sharing (CORS)
- S3-BatchVorgänge
- S3-Bestandsberichte
- Ändern der Bucket-Standard-Verschlüsselung
- Öffentliche Buckets
- Multi-Faktor Authentifizierung (MFA) aktivieren
- S3-Lebenszyklusübergänge (abgesehen von der Objektlöschung und dem Abbrechen unvollständiger mehrteiliger Uploads)
- S3-Objektsperre aufgrund gesetzlicher Aufbewahrungsfristen
- Aufrechterhaltung der Objektsperre
- Serverseitige Verschlüsselung mit Schlüsseln, die von AWS Key Management Service (AWS KMS) (SSE-KMS) verwaltet werden
- S3-Replikationszeitkontrolle (S3 RTC)

- Amazon CloudWatch Anfragemetriken
- Metrik-Konfiguration
- Transfer Acceleration
- S3-Ereignis-Benachrichtigungen
- Buckets mit Zahlung durch den Anforderer
- S3 Select
- AWS Lambda-Ereignisse
- Server access logging (Server-Zugriffsprotokollierung)
- HTTP POST-Anforderungen
- SOAP
- Websitezugriff

Netzwerkanforderungen von S3 on Outposts

- Um Anforderungen an einen Zugriffspunkt für S3 in Outposts weiterzuleiten, müssen Sie einen S3-in-Outposts-Endpunkt erstellen und konfigurieren. Für Endpunkte für S3 in Outposts gelten die folgenden Beschränkungen:
  - Jeder Virtual Private Cloud (VPC) in einem Outpost kann ein Endpunkt zugeordnet sein und Sie können bis zu 100 Endpunkte pro Outpost verwenden.
  - Sie können einem Endpunkt mehrere Zugriffspunkte zuordnen.
  - Endpunkte können Sie nur zu VPCs mit CIDR-Blöcken in den Subspaces der folgenden CIDR-Bereiche hinfügen:
    - 10.0.0.0/8
    - 172.16.0.0/12
    - 192.168.0.0/16
- Endpunkte zu einem Outpost können Sie nur aus VPCs mit nicht überlappenden CIDR-Blöcken erstellen.
- Sie können einen Endpunkt nur aus seinem Outposts-Subnetz erstellen.
- Das Subnetz, das Sie zum Erstellen eines Endpunkts verwenden, muss vier IP-Adressen enthalten, die S3 in Outposts verwenden kann.
- Wenn Sie den kundeneigenen IP-Adresspool (CoIP-Pool) angeben, muss dieser vier IP-Adressen enthalten, die S3 in Outposts verwenden kann.

- Sie können nur einen Endpunkt pro Outpost pro VPC erstellen.

Erste Schritte mit Amazon S3 on Outposts

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (OUTPOSTS). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden.

Mit Amazon S3 on Outposts können Sie die Amazon-S3-APIs und -Funktionen wie Objektspeicherung, Zugriffsrichtlinien, Verschlüsselung und Tagging auf AWS Outposts wie bei Amazon S3 verwenden. Weitere Informationen zu S3 on Outposts finden Sie unter Was ist Amazon S3 on Outposts?.

Themen

- Einrichten von IAM mit S3 on Outposts
- Erste Schritte unter Verwendung der AWS Management Console
- Erste Schritte mit der AWS CLI und dem SDK for Java

Einrichten von IAM mit S3 on Outposts

AWS Identity and Access Management (IAM) ist ein AWS-Service, mit dem ein Administrator den Zugriff auf AWS-Ressourcen sicher steuern kann. IAM-Administratoren steuern, wer authentifiziert (angemeldet) und autorisiert (im Besitz von Berechtigungen) ist, Amazon S3 auf Outpost-Ressourcen zu nutzen. IAM ist ein AWS-Service, den Sie ohne zusätzliche Kosten verwenden können. Standardmäßig haben IAM-Benutzer keine Berechtigungen für S3 auf Outpost-Ressourcen und -Vorgänge. Um Zugriffsberechtigungen für S3 auf Outpost-Ressourcen und API-Operationen zu gewähren, können Sie IAM verwenden, um Benutzer, Gruppen oder Rollen zu erstellen und Berechtigungen zuzuweisen.

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in AWS IAM Identity Center:

  Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter Erstellen eines Berechtigungssatzes im AWS IAM Identity Center-Benutzerhandbuch.
- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden:

  Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter Erstellen einer Rolle für einen externen Identitätsanbieter (Verbund) im IAM-Benutzerhandbuch.
- IAM-Benutzer:
  - Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter Erstellen einer Rolle für einen IAM-Benutzer im IAM-Benutzerhandbuch.
  - (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter Hinzufügen von Berechtigungen zu einem Benutzer (Konsole) im IAM-Benutzerhandbuch.

Zusätzlich zu den IAM-Richtlinien unterstützt S3 on Outposts sowohl Bucket- als auch Zugriffspunkt-Richtlinien. Bucket-Richtlinien und Zugriffspunk-Richtlinien sind ressourcenbasierte Richtlinien, die mit der S3-on-Outposts-Ressource verbunden sind.

- Eine Bucket-Richtlinie ist mit dem Bucket verknüpft und erlaubt oder verweigert Anfragen an den Bucket und die darin enthaltenen Objekte auf der Grundlage der Elemente in der Richtlinie.
- Im Gegensatz dazu ist eine Zugriffspunkt-Richtlinie mit dem Zugriffspunkt verbunden und erlaubt oder verweigert Anfragen an den Zugriffspunkt.

Die Zugriffspunkt-Richtlinie funktioniert mit der Bucket-Richtlinie, die dem zugrunde liegenden S3-on-Outposts-Bucket zugeordnet ist. Damit eine Anwendung oder ein Benutzer über einen S3-on-Outposts-Zugriffspunkt auf Objekte in einem S3-on-Outposts-Bucket zugreifen kann, müssen sowohl die Zugriffspunkt- als auch die Bucket-Richtlinie die Anfrage zulassen.

Einschränkungen, die Sie in eine Zugriffspunktrichtlinie einschließen, gelten nur für Anforderungen, die über diesen Zugriffspunkt eingehen. Wenn beispielsweise ein Zugriffspunkt mit einem Bucket verbunden ist, können Sie die Zugriffspunkt-Richtlinie nicht verwenden, um Anfragen, die direkt an den Bucket gerichtet sind, zuzulassen oder zu verweigern. Einschränkungen, die Sie auf eine Bucket-Richtlinie anwenden, können jedoch Anfragen zulassen oder verweigern, die direkt an den Bucket oder über den Zugriffspunkt gestellt werden.

In einer IAM-Richtlinie oder einer ressourcenbasierten Richtlinie legen Sie fest, welche S3-on-Outposts-Aktionen erlaubt oder abgelehnt werden sollen. S3 on Outposts-Aktionen entsprechen spezifischen S3-on-Outposts-API-Operationen. Aktionen von S3 on Outposts verwenden das Namespace-Präfix `s3-outposts:`. Anfragen, die in einer AWS-Region an die Steuerungs-API von S3 on Outposts gesendet werden, sowie Anfragen, die an die Objekt-API-Endpunkte im Outpost gesendet werden, werden mit IAM authentifiziert und anhand des Namespace-Präfixes `s3-outposts:` autorisiert. Zur Zusammenarbeit mit S3 on Outposts konfigurieren Sie Ihre IAM-Benutzer und autorisieren diese anhand des IAM-Namespace für `s3-outposts:`.

Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 on Outposts in der Service-Autorisierungs-Referenz.

> **Note**
>
> - Zugriffssteuerungslisten (ACLs) werden von S3 on Outposts nicht unterstützt.
> - S3 on Outposts setzt standardmäßig den Bucket-Besitzer als Objekteigentümer ein, um sicherzustellen, dass der Eigentümer eines Buckets nicht am Zugriff auf oder am Löschen von Objekten gehindert werden kann.
> - In S3 on Outposts ist S3 Block Public Access stets aktiviert, um sicherzustellen, dass nie öffentlich auf Objekte zugegriffen werden kann.

Weitere Informationen zur Einrichtung von IAM für S3 on Outposts finden Sie in den folgenden Themen.

Themen

Prinzipale für die Richtlinien von S3 on Outposts

Wenn Sie eine ressourcenbasierte Richtlinie erstellen, um Zugriff auf Ihren S3-on-Outposts-Bucket zu gewähren, müssen Sie das `Principal`-Element verwenden, um die Person oder Anwendung

anzugeben, die eine Anfrage für eine Aktion oder einen Vorgang auf dieser Ressource stellen kann. Für S3-on-Outposts-Richtlinien können Sie einen der folgenden Prinzipals verwenden:

- Ein AWS-Konto
- Ein IAM-Benutzer
- Eine IAM-Rolle
- Alle Prinzipale durch Angabe eines Platzhalters (*) in einer Richtlinie, die ein `Condition`-Element zur Beschränkung des Zugriffs auf einen bestimmten IP-Bereich verwendet

> **Important**
>
> Sie können keine Richtlinie für einen S3-on-Outposts-Bucket schreiben, die einen Platzhalter (*) im `Principal`-Element verwendet, es sei denn, die Richtlinie enthält auch eine `Condition`, die den Zugriff auf einen bestimmten IP-Bereich beschränkt. Mit dieser Beschränkung wird sichergestellt, dass es keinen öffentlichen Zugriff auf Ihren S3-on-Outposts-Bucket gibt. Ein Beispiel finden Sie unter Beispielrichtlinien für S3 on Outposts.

Weitere Informationen zu den `Principal`-Element finden Sie unter AWS-JSON-Richtlinienelemente: Prinzipal im IAM-Benutzerhandbuch.

Ressourcen-ARNs für S3 on Outposts

Amazon-Ressourcennamen (ARNs) für S3 on Outposts enthalten zusätzlich zur AWS-Region, in der sich der Outpost befindet, die AWS-Konto-ID und den Ressourcennamen. Wenn Sie auf Ihre Outposts-Buckets und -Objekte zugreifen und Aktionen für diese ausführen möchten, müssen Sie eines der ARN-Formate verwenden, die in der folgenden Tabelle aufgeführt sind.

Der *`partition`*-Wert im ARN bezieht sich auf eine Gruppe von AWS-Regionen. Jedes AWS-Konto ist auf eine Partition ausgelegt. Im Folgenden werden die unterstützten Partitionen angezeigt:

- `aws` – AWS-Regionen
- `aws-us-gov` – AWS GovCloud (US)-Regionen

ARN-Formate für S3 on Outposts

ARN für Amazon S3 on Outposts	ARN-Format	Beispiel
Bucket-ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /bucket/*bucket_name*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d28a6a232904* /bucket/*DOC-EXAMPLE-BUCKET1*
Zugriffspunkt-ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /accesspoint/ *accesspoint_name*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d28a6a232904* /accesspoint/ *access-point-name*
Objekt-ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /bucket/*bucket_name* /object/*object_key*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d28a6a232904* /bucket/*DOC-EXAMPLE-BUCKET1 /object/myobject*
ARN des Zugriffspunktobjekts in S3 on Outposts (wird in Richtlinien verwendet)	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /accesspoint/ *accesspoint_name* /object/*object_key*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d28a6a232904* /accesspoint/ *access-point-name/object/myobject*

ARN für Amazon S3 on Outposts	ARN-Format	Beispiel
ARN für S3 on Outposts	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d28a6a232904*

Beispielrichtlinien für S3 on Outposts

Example : S3-on-Outposts-Bucket-Richtlinie mit einem AWS-Konto-Prinzipal

Die folgende Bucket-Richtlinie verwendet einen AWS-Konto-Prinzipal, um den Zugriff auf einen S3-on-Outposts-Bucket zu gewähren. Wenn Sie diese Bucket-Richtlinie verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

```
{
   "Version":"2012-10-17",
   "Id":"ExampleBucketPolicy1",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket"
      }
   ]
}
```

Example : S3-on-Outposts-Bucket-Richtlinie mit Platzhalterprinzipal (*****) und Bedingungsschlüssel, um den Zugriff auf einen bestimmten IP-Bereich zu beschränken

Die folgende Bucket-Richtlinie verwendet einen Platzhalterprinzipal (`*`) mit der `aws:SourceIp`-Bedingung, um den Zugriff auf einen bestimmten IP-Bereich zu beschränken. Wenn Sie diese

Bucket-Richtlinie verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre eigenen Informationen.

```
{
    "Version": "2012-10-17",
    "Id": "ExampleBucketPolicy2",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": { "AWS" : "*" },
            "Action":"s3-outposts:*",
            "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket",
            "Condition" : {
                "IpAddress" : {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NotIpAddress" : {
                    "aws:SourceIp": "198.51.100.0/24"
                }
            }
        }
    ]
}
```

Berechtigungen für S3-on-Outposts-Endpunkte

S3 on Outposts erfordert eigene Berechtigungen in IAM, um S3-on-Outposts-Endpunktaktionen zu verwalten.

> **Note**
>
> - Für Endpunkte, die den Zugriffstyp des kundeneigenen IP-Adresspools (CoIP-Pool) verwenden, müssen Sie außerdem über Berechtigungen zum Arbeiten mit IP-Adressen aus Ihrem CoIP-Pool verfügen, wie in der folgenden Tabelle beschrieben.
> - Bei freigegebenen Konten, die mit AWS Resource Access Manager auf S3 auf Outposts zugreifen, können Benutzer dieser freigegebenen Konten keine eigenen Endpunkte in einem freigegebenen Subnetz erstellen. Wenn ein Benutzer in einem freigegebenen Konto

seine eigenen Endpunkte verwalten möchte, muss das freigegebene Konto ein eigenes Subnetz in Outposts erstellen. Weitere Informationen finden Sie unter the section called “Freigabe von S3 on Outposts”.

S3-on-Outposts-Endpunkt-bezogene IAM-Berechtigungen

Action	IAM-Berechtigungen
`CreateEndpoint`	`s3-outposts:CreateEndpoint` `ec2:CreateNetworkInterface` `ec2:DescribeNetworkInterfaces` `ec2:DescribeVpcs` `ec2:DescribeSecurityGroups` `ec2:DescribeSubnets` `ec2:CreateTags` `iam:CreateServiceLinkedRole` Für Endpunkte, die den Zugriffstyp des kundeneigenen On-Premises-IP-Adresspools (CoIP-Pool) verwenden, sind die folgenden zusätzlichen Berechtigungen erforderlich: `s3-outposts:CreateEndpoint` `ec2:DescribeCoipPools` `ec2:GetCoipPoolUsage` `ec2:AllocateAddress` `ec2:AssociateAddress` `ec2:DescribeAddresses`

Action	IAM-Berechtigungen
	`ec2:DescribeLocalGatewayRouteTableVpcAssociations`
`DeleteEndpoint`	`s3-outposts:DeleteEndpoint` `ec2:DeleteNetworkInterface` `ec2:DescribeNetworkInterfaces` Für Endpunkte, die den Zugriffstyp des kundeneigenen On-Premises-IP-Adresspools (CoIP-Pool) verwenden, sind die folgenden zusätzlichen Berechtigungen erforderlich: `s3-outposts:DeleteEndpoint` `ec2:DisassociateAddress` `ec2:DescribeAddresses` `ec2:ReleaseAddress`
`ListEndpoints`	`s3-outposts:ListEndpoints`

> **Note**
>
> Sie können Ressourcen-Markierungen in einer IAM-Richtlinie verwenden, um Berechtigungen zu verwalten.

Serviceverknüpfte Rollen für S3 on Outposts

S3 on Outposts verwendet mit dem IAM-Service verknüpfte Rollen, um einige Netzwerkressourcen in Ihrem Namen zu erstellen. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für S3 on Outposts.

Erste Schritte unter Verwendung der AWS Management Console

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (OUTPOSTS). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Weitere Informationen zu den ersten Schritten mit S3 on Outposts unter Verwendung der Konsole finden Sie in den folgenden Themen. Informationen zu den ersten Schritten unter Verwendung der AWS CLI oder AWS SDK for Java finden Sie unter Erste Schritte mit der AWS CLI und dem SDK for Java.

Themen

Einen Bucket, einen Zugriffspunkt und einen Endpunkt erstellen

Die folgende Vorgehensweise veranschaulicht, wie Sie Ihren ersten Bucket in S3 on Outposts erstellen können. Wenn Sie einen Bucket mit der Konsole erstellen, erstellen Sie auch einen Zugriffspunkt und einen Endpunkt, die mit dem Bucket verknüpft sind, sodass Sie sofort mit dem Speichern von Objekten in Ihrem Bucket beginnen können.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie Outposts-Bucket erstellen.
4. Geben Sie unter Bucket name (Bucket-Name) einen DNS-kompatiblen Namen für Ihren Bucket ein.

Der Bucket-Name muss ...:

- innerhalb des AWS-Konto, des Outposts und der AWS-Region, in der sich der Outpost befindet, eindeutig sein.
- Er muss zwischen 3 und 63 Zeichen lang sein.
- Enthält keine Großbuchstaben.
- mit einem Kleinbuchstaben oder einer Zahl beginnen.

Der Name eines einmal erstellter Buckets kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

> **Important**
>
> Vermeiden Sie, vertrauliche Informationen, wie Kontonummern, in den Bucket-Namen einzubeziehen. Der Bucket-Name wird in der URL angezeigt, die auf die Objekte im Bucket verweist.

5. Wählen Sie unter Outpost den Outpost aus, in dem sich der Bucket befinden soll.
6. Legen Sie unter Bucket Versioning (Bucket-Versionsverwaltung) den S3-Versionsverwaltungsstatus für Ihren S3-on-Outposts-Bucket auf eine der folgenden Optionen fest:

   - Disable (Deaktivieren) (Standard) – Der Bucket wird nicht versioniert.
   - Enable (Aktivieren) – Aktiviert die S3-Versionsverwaltung für die Objekte im Bucket. Alle Objekte, die dem Bucket hinzugefügt werden, erhalten eine eindeutige Versions-ID.

   Weitere Informationen über das S3-Versioning finden Sie unter Verwalten der S3-Versionierung für Ihren S3-on-Outposts-Bucket.
7. (Optional) Fügen Sie ggf. optional tags (optionale Markierungen) hinzu, die Sie mit dem Outposts-Bucket verknüpfen möchten. Sie können Markierungen nutzen, um Kriterien für einzelne Projekte oder Gruppen von Projekten nachzuverfolgen oder um Ihre Buckets unter Verwendung der Kostenzuordnungs-Markierungen zu kennzeichnen.

   Standardmäßig werden alle in Ihrem Outposts-Bucket gespeicherten Objekte mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3)

gespeichert. Sie können auch explizit auswählen, Objekte mithilfe serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) zu speichern. Zum Ändern des Verschlüsselungstyps müssen Sie die REST-API, die AWS Command Line Interface (AWS CLI) oder AWS-SDKs verwenden.

8. Geben Sie im Abschnitt Einstellungen für den Zugriffspunkt für Outposts den Namen des Zugriffspunkts ein.

   S3-on-Outposts-Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in S3 on Outposts. Zugriffspunkte sind benannte Netzwerkendpunkte, die Outposts-Buckets zugeordnet sind, mit denen Sie S3-Objektoperationen ausführen können. Weitere Informationen finden Sie unter Zugriffspunkte.

   Zugriffspunkts-Namen müssen innerhalb des Kontos für diese Region und diesen Outpost eindeutig sein und den entspreche Einschränkungen von Access Points.

9. Wählen Sie die VPC für diesen Amazon-S3-on-Outposts-Zugriffspunkt.

   Wenn Sie keine VPC haben, wählen Sie Create VPC (VPC erstellen) aus. Weitere Informationen finden Sie unter Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind.

   Eine Virtual Private Cloud (VPC) ermöglicht es Ihnen, AWS-Ressourcen in einem virtuellen Netzwerk zu launchen, das Sie definieren. Dieses virtuelle Netzwerk entspricht weitgehend einem herkömmlichen Netzwerk, wie Sie es in Ihrem Rechenzentrum betreiben würden, kann jedoch die Vorzüge der skalierbaren Infrastruktur von nutze AWS.

10. (Optional für eine vorhandene VPC) Wählen Sie ein Endpoint subnet (Endpunkt-Subnetz) für Ihren Endpunkt aus.

    Ein Subnetz ist ein Bereich an IP-Adressen in Ihrer VPC. Wenn Sie nicht das gewünschte Subnetz haben, wählen Sie Subnetz erstellen. Weitere Informationen finden Sie unter Vernetzung für S3 on Outposts.

11. (Optional für eine vorhandene VPC) Wählen Sie eine Endpoint security group (Endpunkt-Sicherheitsgruppe) für Ihren Endpunkt aus.

    Eine Sicherheitsgruppe dient als virtuelle Firewall zur Steuerung von ein- und ausgehendem Datenverkehr.

12. (Optional für eine vorhandene VPC) Wählen Sie den Endpoint access type (Endpunktzugriffstyp) aus:

    - Privat – Zur Verwendung mit der VPC.

- IP im Besitz des Kunden – Zur Verwendung mit einem kundeneigenen IP-Adresspool (CoIP-Pool) Ihres On-Premises-Netzwerks.

13. (Optional) Geben Sie die Outpost access point policy (Outpost-Zugriffspunkt-Richtlinie) an. Die Konsole zeigt automatisch den Amazon-Ressourcennamen (ARN) für den Zugriffspunkt an, den Sie in der Richtlinie verwenden können.
14. Wählen Sie Outposts-Bucket erstellen.

> **Note**
>
> Es kann bis zu 5 Minuten dauern, bis der Outpost-Endpunkt erstellt und der Bucket einsatzbereit ist. Um zusätzliche Bucket-Einstellungen zu konfigurieren, wählen Sie Details anzeigen.

Nächste Schritte

Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Nachdem Sie einen S3-on-Outposts-Bucket, einen Zugriffspunkt und einen Endpunkt erstellt haben, können Sie die AWS CLI oder das SDK für Java verwenden, um ein Objekt in Ihren Bucket hochzuladen. Weitere Informationen finden Sie unter Schritt 4: Hochladen eines Objekts in einen S3-on-Outposts-Bucket.

Erste Schritte mit der AWS CLI und dem SDK for Java

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (`OUTPOSTS`). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei

Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Für die ersten Schritte mit S3 on Outposts müssen Sie einen Bucket, einen Zugriffspunkt und einen Endpunkt erstellen. Anschließend können Sie Ihre Objekte in den Bucket hochladen. Die folgenden Beispiele veranschaulichen, wie Sie die ersten Schritte mit S3 on Outposts mithilfe der AWS CLI und des SDK for Java ausführen können. Die ersten Schritte mit der Konsole finden Sie unter Erste Schritte unter Verwendung der AWS Management Console.

Themen

Schritt 1: Erstellen eines Buckets

Die folgenden Beispiele für AWS CLI und SDK für Java veranschaulichen, wie Sie einen S3-on-Outposts-Bucket erstellen.

AWS CLI

Example

Im folgenden Beispiel wird ein S3-on-Outposts-Bucket (`s3-outposts:CreateBucket`) mithilfe der AWS CLI erstellt. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

SDK for Java

Example

Im folgenden Beispiel wird ein S3-on-Outposts-Bucket (`s3-outposts:CreateBucket`) mit dem SDK für Java erstellt.

```
import com.amazonaws.services.s3control.model.*;

public String createBucket(String bucketName) {

    CreateBucketRequest reqCreateBucket = new CreateBucketRequest()
            .withBucket(bucketName)
            .withOutpostId(OutpostId)
            .withCreateBucketConfiguration(new CreateBucketConfiguration());

    CreateBucketResult respCreateBucket =
 s3ControlClient.createBucket(reqCreateBucket);
    System.out.printf("CreateBucket Response: %s%n", respCreateBucket.toString());

    return respCreateBucket.getBucketArn();

}
```

Schritt 2: Erstellen eines Zugriffspunkts

Um auf Ihren Amazon-S3-on-Outposts-Bucket zuzugreifen, müssen Sie einen Zugriffspunkt erstellen und konfigurieren. Diese Beispiele veranschaulichen, wie Sie einen Zugriffspunkt mithilfe der AWS CLI und des SDK for Java erstellen.

Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie Amazon S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Zugriffspunkte unterstützen ausschließlich die virtuelle, Host-artige Adressierung.

AWS CLI

Example

Im folgenden AWS CLI-Beispiel wird ein Zugriffspunkt für einen Outposts-Bucket erstellt. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
```

```
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

SDK for Java

Example

Im folgenden Beispiel für SDK für Java wird ein Zugriffspunkt für einen Outposts-Bucket erstellt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
import com.amazonaws.services.s3control.model.*;

public String createAccessPoint(String bucketArn, String accessPointName) {

    CreateAccessPointRequest reqCreateAP = new CreateAccessPointRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withName(accessPointName)
            .withVpcConfiguration(new VpcConfiguration().withVpcId("vpc-12345"));

    CreateAccessPointResult respCreateAP =
 s3ControlClient.createAccessPoint(reqCreateAP);
    System.out.printf("CreateAccessPoint Response: %s%n", respCreateAP.toString());

    return respCreateAP.getAccessPointArn();

}
```

Schritt 3: Erstellen eines Endpunkts

Wenn Sie Anforderungen an einen Zugriffspunkt für Amazon S3 on Outposts weiterleiten möchten, müssen Sie einen S3-on-Outposts-Endpunkt erstellen und konfigurieren. Zum Erstellen eines Endpunkts muss Ihr Service-Link eine aktive Verbindung mit Ihrer Outposts-Heimatregion aufweisen. Jeder Virtual Private Cloud (VPC) in Ihrem Outpost kann ein Endpunkt zugeordnet sein. Weitere Informationen zu Endpunktkontingenten finden Sie unter Netzwerkanforderungen von S3 on Outposts. Sie müssen einen Endpunkt erstellen, um auf Ihre Outposts-Buckets zugreifen und Objekt-Operationen ausführen zu können. Weitere Informationen finden Sie unter Endpunkte.

Diese Beispiele veranschaulichen, wie Sie einen Endpunkt mithilfe der AWS CLI und des SDK for Java erstellen. Weitere Informationen zu den erforderlichen Berechtigungen für das Erstellen und Verwalten von Endpunkten finden Sie unter Berechtigungen für S3-on-Outposts-Endpunkte.

AWS CLI

Example

Im folgenden AWS CLI-Beispiel wird ein Endpunkt für einen Outpost mithilfe des VPC-Ressourcenzugriffstyps erstellt. Die VPC ist vom Subnetz abgeleitet. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

Im folgenden AWS CLI-Beispiel wird ein Endpunkt für einen Outpost mit dem Zugriffstyp des kundeneigenen IP-Adresspools (CoIP-Pool) erstellt. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

SDK for Java

Example

Im folgenden SDK für Java-Beispiel wird ein Endpunkt für einen Outpost erstellt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.CreateEndpointRequest;
import com.amazonaws.services.s3outposts.model.CreateEndpointResult;

public void createEndpoint() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    CreateEndpointRequest createEndpointRequest = new CreateEndpointRequest()
```

```
                .withOutpostId("op-0d79779cef3c30a40")
                .withSubnetId("subnet-8c7a57c5")
                .withSecurityGroupId("sg-ab19e0d1")
                .withAccessType("CustomerOwnedIp")
                .withCustomerOwnedIpv4Pool("ipv4pool-coip-12345678901234567");
    // Use .withAccessType and .withCustomerOwnedIpv4Pool only when the access type
 is
    // customer-owned IP address pool (CoIP pool)
    CreateEndpointResult createEndpointResult =
 s3OutpostsClient.createEndpoint(createEndpointRequest);
    System.out.println("Endpoint is created and its ARN is " +
 createEndpointResult.getEndpointArn());
}
```

Schritt 4: Hochladen eines Objekts in einen S3-on-Outposts-Bucket

Objekte sind die Grundeinheiten, die in Amazon S3 on Outposts gespeichert sind. Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Amazon-Ressourcennamen (ARN) oder den Alias des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Objekte, das den AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, Outpost-ID, Bucket-Name und den Namen des Zugriffspunkts umfasst:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Die folgenden Beispiele für die AWS CLI und AWS SDK for Java veranschaulichen, wie Sie ein Objekt unter Verwendung eines Zugriffspunkts in einen S3-on-Outposts-Bucket hochladen.

AWS CLI

Example

Im folgenden Beispiel wird ein Objekt mit dem Namen `sample-object.xml` in einen S3-on-Outposts-Bucket (`s3-outposts:PutObject`) mit der AWS CLI eingefügt. Zum Verwenden dieses Befehls ersetzen Sie *`user input placeholder`* durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter put-object in der AWS CLI-Referenz.

```
aws s3api put-object --bucket arn:aws:s3-
outposts:Region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml --body sample-object.xml
```

SDK for Java

Example

Im folgenden Beispiel wird ein Objekt mithilfe des SDK for Java in einen S3-on-Outposts-Bucket eingefügt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie jeden *`user input placeholder`* durch Ihre Informationen. Weitere Informationen finden Sie unter Objekte hochladen.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectMetadata;
import com.amazonaws.services.s3.model.PutObjectRequest;

import java.io.File;

public class PutObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String stringObjKeyName = "*** String object key name ***";
        String fileObjKeyName = "*** File object key name ***";
        String fileName = "*** Path to file to upload ***";
```

```
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Upload a text string as a new object.
            s3Client.putObject(accessPointArn, stringObjKeyName, "Uploaded String
 Object");

            // Upload a file as a new object with ContentType and title specified.
            PutObjectRequest request = new PutObjectRequest(accessPointArn,
 fileObjKeyName, new File(fileName));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentType("plain/text");
            metadata.addUserMetadata("title", "someTitle");
            request.setMetadata(metadata);
            s3Client.putObject(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Vernetzung für S3 on Outposts

Sie können Amazon S3 on Outposts verwenden, um Objekte lokal für Anwendungen zu speichern und abzurufen, die lokalen Datenzugriff, Datenverarbeitung und Datenresidenz erfordern. In diesem Abschnitt werden die Netzwerkanforderungen für den Zugriff auf S3 on Outposts beschrieben.

Themen

- Auswählen des Netzwerkzugriffstyps
- Zugriff auf Ihre S3-on-Outposts-Buckets und -Objekte

- Kontenübergreifende Elastic-Network-Schnittstellen

Auswählen des Netzwerkzugriffstyps

Sie können von einer VPC oder von Ihrem lokalen Netzwerk aus auf S3 on Outposts zugreifen. Sie kommunizieren mit Ihrem Outposts-Bucket über einen Zugriffspunkt und eine Endpunktverbindung. Dadurch bleibt der Datenverkehr zwischen Ihrer VPC und Ihren S3-on-Outposts-Buckets innerhalb des AWS-Netzwerks. Beim Erstellen eines Endpunkts müssen Sie den Endpunktzugriffstyp entweder als `Private` (für VPC-Routing) oder `CustomerOwnedIp` (für einen kundeneigenen IP-Adresspool [CoIP-Pool]) angeben.

- `Private`(für VPC-Routing) – Wenn Sie den Zugriffstyp nicht angeben, verwendet S3 on Outposts standardmäßig `Private`. Mit dem Zugriffstyp `Private` benötigen Instances in Ihrer VPC keine öffentlichen IP-Adressen, um mit Ressourcen in Ihrem Outpost zu kommunizieren. Sie können von einer VPC aus mit S3 on Outposts arbeiten. Der Zugriff auf diesen Endpunkttyp ist über direktes VPC-Routing über Ihr lokales Netzwerk möglich. Weitere Informationen finden Sie unter Roouting-Tabellen für lokale Gateways im AWS-Outposts-Benutzerhandbuch.
- `CustomerOwnedIp`(für CoIP-Pool) – Wenn Sie den Zugriffstyp `Private` nicht standardmäßig verwenden und `CustomerOwnedIp` auswählen, müssen Sie einen IP-Adressbereich angeben. Sie können diesen Zugriffstyp verwenden, um mit S3 on Outposts sowohl aus Ihrem On-Premises-Netzwerk als auch innerhalb einer VPC zu arbeiten. Wenn Sie auf S3 on Outposts innerhalb einer VPC zugreifen, ist Ihr Datenverkehr auf die Bandbreite des lokalen Gateways beschränkt.

Zugriff auf Ihre S3-on-Outposts-Buckets und -Objekte

Um auf Ihre S3 on Outposts-Buckets und -Objekte zugreifen zu können, benötigen Sie Folgendes:

- Ein Zugriffspunkt für die VPC.
- Ein Endpunkt für die gleiche VPC.
- Eine aktive Verbindung zwischen Ihrem Outpost und Ihrer AWS-Region. Weitere Informationen über die Verbindungserstellung Ihres Outposts zu einer Region finden Sie unter Outpost-Konnektivität zu AWS-Regionen im AWS-Outposts-Benutzerhandbuch.

Weitere Informationen zum Zugriff auf Buckets und Objekte in S3 on Outposts finden Sie unter Arbeiten mit S3-on-Outposts-Buckets undArbeiten mit S3-on-Outposts-Objekten.

Kontenübergreifende Elastic-Network-Schnittstellen

S3-on-Outposts-Endpunkte sind benannte Ressourcen mit Amazon-Ressourcennamen (ARNs). Wenn diese Endpunkte erstellt werden, richtet AWS Outposts mehrere kontoübergreifende Elastic-Network-Schnittstellen ein. Die kontenübergreifenden Elastic-Network-Schnittstellen von S3 on Outposts sind wie andere Netzwerkschnittstellen mit einer Ausnahme: S3 on Outposts ordnet die kontenübergreifenden Elastic-Network-Schnittstellen Amazon-EC2-Instances zu.

Das S3-on-Outposts-Domain-Name-System (DNS) verteilt Ihre Anfragen über die kontoübergreifende Elastic-Network-Schnittstelle. S3 on Outposts erstellt die kontoübergreifende Elastic Network-Schnittstelle in Ihrem AWS-Konto, die im Bereich Netzwerkschnittstellen der Amazon EC2-Konsole sichtbar ist.

Für Endpunkte, die den CoIP-Pool-Zugriffstyp verwenden, weist S3 on Outposts IP-Adressen der kontoübergreifenden Elastic-Network-Schnittstelle aus dem konfigurierten CoIP-Pool zu und ordnet sie dieser zu.

Arbeiten mit S3-on-Outposts-Buckets

Mit Amazon S3 on Outposts können Sie S3-Buckets in Ihren AWS Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premises speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (`OUTPOSTS`). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon S3 verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Sie kommunizieren mit Ihrenm Outpost-Buckets über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Für den Zugriff auf Ihre S3-on-Outposts-Buckets und -Objekte benötigen Sie einen Zugriffspunkt für die VPC und einen Endpunkt für dieselbe VPC. Weitere Informationen finden Sie unter Vernetzung für S3 on Outposts.

Buckets

In S3 on Outposts sind Bucket-Namen für einen Outpost eindeutig und erfordern AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, die Outpost ID und den Bucket-Namen, um sie zu identifizieren.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

Weitere Informationen finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Zugriffspunkte

Amazon S3 on Outposts unterstützt reine Virtual-Private-Cloud(VPC)-Zugriffspunkte als einzige Möglichkeit, auf Ihre Outposts-Buckets zuzugreifen.

Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie Amazon S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Zugriffspunkte unterstützen ausschließlich die virtuelle, Host-artige Adressierung.

Im folgenden Beispiel wird das ARN-Format gezeigt, das Sie für S3-on-Outposts-Zugriffspunkte verwenden. Der Zugriffspunkt-ARN umfasst den AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, die Outpost-ID und den Namen des Zugriffspunkts.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Endpunkte

Um Anforderungen an einen Zugriffspunkt für S3 on Outposts weiterzuleiten, müssen Sie einen S3-on-Outposts-Endpunkt erstellen und konfigurieren. Mit S3-on-Outposts-Endpunkten können Sie Ihre VPC privat mit Ihrem Outpost-Bucket verbinden. S3-on-Outposts-Endpunkte sind virtuelle Uniform Resource Identifiers (URIs) des Einstiegspunkts zu Ihrem S3-on-Outposts-Bucket. Es handelt sich bei ihnen um horizontal skalierte, redundante und hochverfügbare VPC-Komponenten.

Jeder Virtual Private Cloud (VPC) in Ihrem Outpost kann ein Endpunkt zugeordnet sein und Sie können bis zu 100 Endpunkte pro Outpost verwenden. Sie müssen diese Endpunkte erstellen, um auf Ihren Outpost-Bucket zugreifen und Objektvorgänge ausführen zu können. Das Erstellen dieser Endpunkte ermöglicht auch, dass das API-Modell und das Verhalten identisch sind, indem dieselben Vorgänge in S3 und S3 on Outposts ausgeführt werden.

API-Vorgänge in S3 on Outposts

Um Outpost-Bucket-API-Vorgänge zu verwalten, hostet S3 on Outposts einen separaten Endpunkt, der sich vom Amazon-S3-Endpunkt unterscheidet. Dieser Endpunkt ist `s3-outposts.`*`region`*`.amazonaws.com`.

Um dieselben Amazon-S3-API-Vorgänge zu verwenden, müssen Sie den Bucket und die Objekte im korrekten ARN-Format signieren. Sie müssen ARNs an API-Vorgänge übergeben, damit Amazon S3 feststellen kann, ob die Anfrage für Amazon S3 (`s3-control.`*`region`*`.amazonaws.com`) oder S3 on Outposts (`s3-outposts.`*`region`*`.amazonaws.com`) gilt. Basierend auf dem ARN-Format kann S3 die Anfrage dann entsprechend signieren und weiterleiten.

Wenn die Anforderung an die Amazon S3-Steuerebene gesendet wird, extrahiert das SDK die Komponenten aus dem ARN und fügt einen zusätzlichen Header `x-amz-`*`outpost-id`* mit dem Wert *`outpost-id`* ein, der aus dem ARN extrahiert wurde. Der Service-Name aus dem ARN wird für die Signierung der Anforderung verwendet, bevor sie an den S3-on-Outposts-Endpunkt weitergeleitet wird. Dieses Verhalten gilt für alle API-Vorgänge, die vom `s3control`-Client verarbeitet werden.

In der folgenden Tabelle sind die fortschrittlichen API-Vorgänge für Amazon S3 on Outposts und ihre Änderungen im Verhältnis zu Amazon S3 aufgeführt.

API	S3-on-Outposts-Parameterwert	
`CreateBucket`	Bucket-Name wie ARN, Outpost-ID	
`ListRegionalBuckets`	Outpost-ID	
`DeleteBucket`	Bucket-Name als ARN	
`DeleteBucketLifecycleConfiguration`	Bucket-Name als ARN	
`GetBucketLifecycleConfiguration`	Bucket-Name als ARN	
`PutBucketLifecycleConfiguration`	Bucket-Name als ARN	
`GetBucketPolicy`	Bucket-Name als ARN	

API	S3-on-Outposts-Parameterwert	
`PutBucketPolicy`	Bucket-Name als ARN	
`DeleteBucketPolicy`	Bucket-Name als ARN	
`GetBucketTagging`	Bucket-Name als ARN	
`PutBucketTagging`	Bucket-Name als ARN	
`DeleteBucketTagging`	Bucket-Name als ARN	
`CreateAccessPoint`	Name des Zugriffspunkts als ARN	
`DeleteAccessPoint`	Name des Zugriffspunkts als ARN	
`GetAccessPoint`	Name des Zugriffspunkts als ARN	
`GetAccessPoint`	Name des Zugriffspunkts als ARN	
`ListAccessPoints`	Name des Zugriffspunkts als ARN	
`PutAccessPointPolicy`	Name des Zugriffspunkts als ARN	
`GetAccessPointPolicy`	Name des Zugriffspunkts als ARN	
`DeleteAccessPointPolicy`	Name des Zugriffspunkts als ARN	

Erstellen und Verwalten von S3 on Outposts-Buckets

Weitere Informationen zum Erstellen und Verwalten von S3-on-Outposts-Buckets finden Sie in den folgenden Themen.

Themen

- Erstellen eines S3-on-Outposts-Buckets
- Hinzufügen von Tags für S3-on-Outposts-Buckets
- Verwalten des Zugriffs auf einen Amazon-S3-on-Outposts-Bucket mit einer Bucket-Richtlinie
- Auflisten von Amazon-S3-on-Outposts-Buckets
- Abrufen eines S3-on-Outposts-Buckets mithilfe der AWS CLI und des SDK for Java

- Löschen Ihres Amazon-S3-on-Outposts-Buckets
- Arbeiten mit Zugriffspunkten von Amazon S3 on Outposts
- Arbeiten mit Amazon-S3-on-Outposts-Endpunkten

Erstellen eines S3-on-Outposts-Buckets

Mit Amazon S3 on Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (`OUTPOSTS`). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

> **Note**
>
> Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das ihm Aktionen zuweisen kann. Buckets verfügen über Konfigurationseigenschaften wie Outpost, Tags, Standard-Verschlüsselung und Zugriffspunkteinstellungen. Zu den Zugriffspunkteinstellungen gehören die Virtual Private Cloud (VPC), die Zugriffspunkt-Richtlinie für den Zugriff auf die Objekte im Bucket sowie andere Metadaten. Weitere Informationen finden Sie unter Spezifikationen für S3 auf Outposts.
>
> Wenn Sie einen Bucket erstellen möchten, der AWS PrivateLink verwendet, um in Ihrer Virtual Private Cloud (VPC) über Schnittstellen-VPC-Endpunkte Zugriff auf die Bucket- und Endpunkt-Verwaltung bereitzustellen, rufen Sie AWS PrivateLink für S3 auf Outposts auf.

Die folgenden Beispiele zeigen, wie Sie einen S3-on-Outposts-Bucket mithilfe der AWS Management Console, der AWS Command Line Interface (AWS CLI) und AWS SDK for Java erstellen.

Verwenden der S3-Konsole

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie Outposts-Bucket erstellen.
4. Geben Sie unter Bucket name (Bucket-Name) einen DNS-kompatiblen Namen für Ihren Bucket ein.

   Der Bucket-Name ...:

   - innerhalb des AWS-Konto, des Outposts und der AWS-Region, in der sich der Outpost befindet, eindeutig sein.
   - Er muss zwischen 3 und 63 Zeichen lang sein.
   - Enthält keine Großbuchstaben.
   - mit einem Kleinbuchstaben oder einer Zahl beginnen.

   Der Name eines einmal erstellter Buckets kann nicht nachträglich geändert werden. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Benennung von Buckets.

   > **Important**
   >
   > Vermeiden Sie, vertrauliche Informationen, wie Kontonummern, in den Bucket-Namen einzubeziehen. Der Bucket-Name wird in der URL angezeigt, die auf die Objekte im Bucket verweist.

5. Wählen Sie unter Outpost den Outpost aus, in dem sich der Bucket befinden soll.
6. Legen Sie unter Bucket Versioning (Bucket-Versionsverwaltung) den S3-Versionsverwaltungsstatus für Ihren S3-on-Outposts-Bucket auf eine der folgenden Optionen fest:

   - Disable (Deaktivieren) (Standard) – Der Bucket wird nicht versioniert.
   - Enable (Aktivieren) – Aktiviert die S3-Versionsverwaltung für die Objekte im Bucket. Alle Objekte, die dem Bucket hinzugefügt werden, erhalten eine eindeutige Versions-ID.

Weitere Informationen über das S3-Versioning finden Sie unter Verwalten der S3-Versionierung für Ihren S3-on-Outposts-Bucket.

7. (Optional) Fügen Sie ggf. optional tags (optionale Markierungen) hinzu, die Sie mit dem Outposts-Bucket verknüpfen möchten. Sie können Markierungen nutzen, um Kriterien für einzelne Projekte oder Gruppen von Projekten nachzuverfolgen oder um Ihre Buckets unter Verwendung der Kostenzuordnungs-Markierungen zu kennzeichnen.

   Standardmäßig werden alle in Ihrem Outposts-Bucket gespeicherten Objekte mit serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) gespeichert. Sie können auch explizit auswählen, Objekte mithilfe serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) zu speichern. Zum Ändern des Verschlüsselungstyps müssen Sie die REST-API, die AWS Command Line Interface (AWS CLI) oder AWS-SDKs verwenden.

8. Geben Sie im Abschnitt Einstellungen für den Zugriffspunkt für Outposts den Namen des Zugriffspunkts ein.

   S3-on-Outposts-Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in S3 on Outposts. Zugriffspunkte sind benannte Netzwerkendpunkte, die Outposts-Buckets zugeordnet sind, mit denen Sie S3-Objektoperationen ausführen können. Weitere Informationen finden Sie unter Zugriffspunkte.

   Zugriffspunkts-Namen müssen innerhalb des Kontos für diese Region und diesen Outpost eindeutig sein und den entspreche Einschränkungen von Access Points.

9. Wählen Sie die VPC für diesen Amazon-S3-on-Outposts-Zugriffspunkt.

   Wenn Sie keine VPC haben, wählen Sie Create VPC (VPC erstellen) aus. Weitere Informationen finden Sie unter Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind.

   Eine Virtual Private Cloud (VPC) ermöglicht es Ihnen, AWS-Ressourcen in einem virtuellen Netzwerk zu launchen, das Sie definieren. Dieses virtuelle Netzwerk entspricht weitgehend einem herkömmlichen Netzwerk, wie Sie es in Ihrem Rechenzentrum betreiben würden, kann jedoch die Vorzüge der skalierbaren Infrastruktur von nutze AWS.

10. (Optional für eine vorhandene VPC) Wählen Sie ein Endpoint subnet (Endpunkt-Subnetz) für Ihren Endpunkt aus.

Ein Subnetz ist ein Bereich an IP-Adressen in Ihrer VPC. Wenn Sie nicht das gewünschte Subnetz haben, wählen Sie Subnetz erstellen. Weitere Informationen finden Sie unter Vernetzung für S3 on Outposts.

11. (Optional für eine vorhandene VPC) Wählen Sie eine Endpoint security group (Endpunkt-Sicherheitsgruppe) für Ihren Endpunkt aus.

    Eine Sicherheitsgruppe dient als virtuelle Firewall zur Steuerung von ein- und ausgehendem Datenverkehr.

12. (Optional für eine vorhandene VPC) Wählen Sie den Endpoint access type (Endpunktzugriffstyp) aus:

    - Privat – Zur Verwendung mit der VPC.
    - IP im Besitz des Kunden – Zur Verwendung mit einem kundeneigenen IP-Adresspool (CoIP-Pool) Ihres On-Premises-Netzwerks.

13. (Optional) Geben Sie die Outpost access point policy (Outpost-Zugriffspunkt-Richtlinie) an. Die Konsole zeigt automatisch den Amazon-Ressourcennamen (ARN) für den Zugriffspunkt an, den Sie in der Richtlinie verwenden können.
14. Wählen Sie Outposts-Bucket erstellen.

> **Note**
>
> Es kann bis zu 5 Minuten dauern, bis der Outpost-Endpunkt erstellt und der Bucket einsatzbereit ist. Um zusätzliche Bucket-Einstellungen zu konfigurieren, wählen Sie Details anzeigen.

Verwendung von AWS CLI

Example

Im folgenden Beispiel wird ein S3-on-Outposts-Bucket (`s3-outposts:CreateBucket`) mithilfe der AWS CLI erstellt. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

Verwenden des AWS-SDKs für Java

Example

Im folgenden Beispiel wird ein S3-on-Outposts-Bucket (`s3-outposts:CreateBucket`) mit dem SDK für Java erstellt.

```
import com.amazonaws.services.s3control.model.*;

public String createBucket(String bucketName) {

    CreateBucketRequest reqCreateBucket = new CreateBucketRequest()
            .withBucket(bucketName)
            .withOutpostId(OutpostId)
            .withCreateBucketConfiguration(new CreateBucketConfiguration());

    CreateBucketResult respCreateBucket =
 s3ControlClient.createBucket(reqCreateBucket);
    System.out.printf("CreateBucket Response: %s%n", respCreateBucket.toString());

    return respCreateBucket.getBucketArn();

}
```

Hinzufügen von Tags für S3-on-Outposts-Buckets

Sie können Tags für Ihre Amazon-S3-on-Outposts-Buckets hinzufügen, um die Speicherkosten oder andere Kriterien für einzelne Projekte oder Gruppen von Projekten zu verfolgen.

> **Note**
>
> Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das seine Markierungen ändern kann.

Verwenden der S3-Konsole

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.

3. Wählen Sie den Outposts-Bucket aus, dessen Tags Sie bearbeiten möchten.
4. Wählen Sie die Registerkarte Properties (Eigenschaften) aus.
5. Wählen Sie unter Tags, die Option Edit (Bearbeiten) aus.
6. (Optional) Wählen Sie für jede Markierung Add new tag (Neue Markierung hinzufügen) und geben Sie den Schlüsselnamen unter Key (Schlüssel) und den Wert unter Value (Wert) ein.

   Fügen Sie alle Tags hinzu, die Sie mit einem Outposts-Bucket verknüpfen möchten, um andere Kriterien für einzelne Projekte oder Gruppen von Projekten zu verfolgen.
7. Wählen Sie Save Changes (Änderungen speichern).

Verwendung der AWS CLI

Das folgende AWS CLI-Beispiel wendet eine Markierungskonfiguration auf einen S3-on-Outposts-Bucket an, wobei ein JSON-Dokument im aktuellen Ordner verwendet wird, das Tags (*tagging.json*) angibt. Zum Verwenden dieses Beispiels ersetzen Sie jeden *user input placeholder* durch Ihre eigenen Informationen.

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --tagging file://tagging.json

tagging.json

{
   "TagSet": [
     {
       "Key": "organization",
       "Value": "marketing"
     }
   ]
}
```

Das folgende AWS CLI-Beispiel wendet eine Markierungskonfiguration direkt von der Befehlszeile aus auf einen S3-on-Outposts-Bucket an.

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --tagging 'TagSet=[{Key=organization,Value=marketing}]'
```

Weitere Informationen über diesen Befehl finden Sie unter put-bucket-tagging in der AWS CLI-Referenz.

Verwalten des Zugriffs auf einen Amazon-S3-on-Outposts-Bucket mit einer Bucket-Richtlinie

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende AWS Identity and Access Management-(IAM)-Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt. Weitere Informationen finden Sie unter Bucket-Richtlinie.

Sie können Ihre Bucket-Richtlinie aktualisieren, um den Zugriff auf Ihren Amazon-S3-on-Outposts-Bucket zu verwalten. Weitere Informationen finden Sie unter den folgenden Themen.

Themen

- Hinzufügen oder Bearbeiten einer Bucket-Richtlinie für einen Amazon-S3-on-Outposts-Bucket
- Anzeigen der Bucket-Richtlinie für Ihren Amazon-S3-on-Outposts-Bucket
- Löschen der Bucket-Richtlinie für Ihren Amazon-S3-on-Outposts-Bucket
- Beispiele für Bucket-Richtlinien

Hinzufügen oder Bearbeiten einer Bucket-Richtlinie für einen Amazon-S3-on-Outposts-Bucket

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende AWS Identity and Access Management-(IAM)-Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt. Weitere Informationen finden Sie unter Bucket-Richtlinie.

In den folgenden Themen erfahren Sie, wie Sie Ihre Bucket-Richtlinie für Amazon S3 on Outposts mithilfe der AWS Management Console, der AWS Command Line Interface (AWS CLI) oder AWS SDK for Java aktualisieren.

Verwenden der S3-Konsole

Eine Bucket-Richtlinie erstellen oder bearbeiten

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, dessen Bucket-Richtlinie Sie bearbeiten möchten.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen).
5. Wählen Sie im Abschnitt Outposts bucket policy (Outposts-Bucket-Richtlinie) die Option Edit (Bearbeiten) aus, um eine neue Richtlinie zu erstellen oder zu bearbeiten.

   Sie können nun die S3-on-Outposts-Bucket-Richtlinie hinzufügen oder bearbeiten. Weitere Informationen finden Sie unter Einrichten von IAM mit S3 on Outposts.

Verwendung der AWS CLI

Im folgenden AWS CLI-Beispiel wird eine Richtlinie für einen Outposts-Bucket eingerichtet.

1. Speichern Sie die folgende Bucket-Richtlinie in einer JSON-Datei. In diesem Beispiel heißt die Datei `policy1.json`. Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
{
   "Version":"2012-10-17",
   "Id":"testBucketPolicy",
   "Statement":[
      {
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket"
      }
   ]
}
```

2. Senden Sie die JSON-Datei als Teil des CLI-Befehls `put-bucket-policy`. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control put-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --policy file://policy1.json
```

Verwenden des AWS-SDKs für Java

Im folgenden SDK für Java-Beispiel wird eine Richtlinie für einen Outposts-Bucket gesetzt.

```
import com.amazonaws.services.s3control.model.*;

public void putBucketPolicy(String bucketArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testBucketPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId+ "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + bucketArn + "\"}]}";

    PutBucketPolicyRequest reqPutBucketPolicy = new PutBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withPolicy(policy);

    PutBucketPolicyResult respPutBucketPolicy =
 s3ControlClient.putBucketPolicy(reqPutBucketPolicy);
    System.out.printf("PutBucketPolicy Response: %s%n",
 respPutBucketPolicy.toString());

}
```

Anzeigen der Bucket-Richtlinie für Ihren Amazon-S3-on-Outposts-Bucket

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende AWS Identity and Access Management-(IAM)-Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt. Weitere Informationen finden Sie unter Bucket-Richtlinie.

In den folgenden Themen erfahren Sie, wie Sie Ihre Bucket-Richtlinie für Amazon S3 on Outposts mithilfe der AWS Management Console, der AWS Command Line Interface (AWS CLI) oder AWS SDK for Java anzeigen.

Verwenden der S3-Konsole

Eine Bucket-Richtlinie erstellen oder bearbeiten

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, dessen Berechtigung Sie bearbeiten möchten.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen) aus.
5. Im Abschnitt Outposts bucket policy (Outposts-Bucket-Richtlinie) können Sie Ihre vorhandene Bucket-Richtlinie überprüfen. Weitere Informationen finden Sie unter Einrichten von IAM mit S3 on Outposts.

Verwendung der AWS CLI

Im folgenden AWS CLI-Beispiel wird eine Richtlinie für einen Outposts-Bucket erhalten. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control get-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

Verwenden des AWS-SDKs für Java

Im folgenden SDK für Java-Beispiel wird eine Richtlinie für einen Outposts-Bucket abgerufen.

```
import com.amazonaws.services.s3control.model.*;

public void getBucketPolicy(String bucketArn) {

    GetBucketPolicyRequest reqGetBucketPolicy = new GetBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    GetBucketPolicyResult respGetBucketPolicy =
 s3ControlClient.getBucketPolicy(reqGetBucketPolicy);
```

```
    System.out.printf("GetBucketPolicy Response: %s%n",
 respGetBucketPolicy.toString());

}
```

Löschen der Bucket-Richtlinie für Ihren Amazon-S3-on-Outposts-Bucket

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende AWS Identity and Access Management-(IAM)-Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt. Weitere Informationen finden Sie unter Bucket-Richtlinie.

In den folgenden Themen erfahren Sie, wie Sie Ihre Richtlinie für Amazon-S3-on-Outposts-Buckets mithilfe der AWS Management Console oder der AWS Command Line Interface (AWS CLI) anzeigen.

Verwenden der S3-Konsole

Löschen einer Bucket-Richtlinie

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, dessen Berechtigung Sie bearbeiten möchten.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen) aus.
5. Wählen Sie im Bereich Outposts-Bucket-Richtlinie die Option Löschen aus.
6. Bestätigen Sie das Löschen.

Verwendung der AWS CLI

Im folgenden Beispiel wird die Bucket-Richtlinie für einen S3-on-Outposts-Bucket (`s3-outposts:DeleteBucket`) mithilfe der AWS CLI gelöscht. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control delete-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

Beispiele für Bucket-Richtlinien

Mit S3-on-Outposts-Bucket-Richtlinien können Sie den Zugriff auf Objekte in Ihren S3-on-Outposts-Buckets sichern, sodass nur Benutzer mit den entsprechenden Berechtigungen darauf zugreifen können. Sie können sogar verhindern, dass authentifizierte Benutzer ohne die entsprechenden Berechtigungen auf Ihre S3-on-Outposts-Ressourcen zugreifen.

Dieser Abschnitt enthält Beispiele für typische Anwendungsfälle für S3-on-Outposts-Bucket-Richtlinien. Wenn Sie diese Richtlinien testen möchten, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen (z. B. Ihren Bucket-Namen).

Wenn Sie einer Gruppe von Objekten Berechtigungen erteilen oder verweigern möchten, können Sie Platzhalterzeichen für (*) Amazon-Ressourcennamen (ARNs) und andere Werte verwenden. Sie können beispielsweise den Zugriff auf Gruppen von Objekten steuern, die mit einem gemeinsamen Präfixbeginnen oder mit einer bestimmten Erweiterung wie `.html` enden.

Weitere Informationen zur AWS Identity and Access Management (IAM)-Richtliniensprache finden Sie unter Einrichten von IAM mit S3 on Outposts.

> **Note**
>
> Wenn Sie s3outposts Berechtigungen mithilfe der Amazon S3-Konsole testen, müssen Sie zusätzliche Berechtigungen gewähren, die die Konsole benötigt, z. B. `s3outposts:createendpoint`, `s3outposts:listendpoints`usw.

Zusätzliche Ressourcen für die Erstellung von Bucket-Richtlinien

- Eine Liste der IAM-Richtlinienaktionen, Ressourcen und Bedingungsschlüssel, die Sie beim Erstellen einer S3-on-Outposts-Bucket-Richtlinie verwenden können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 on Outposts.
- Hinweise zum Erstellen Ihrer S3-on-Outposts-Richtlinie finden Sie unter Hinzufügen oder Bearbeiten einer Bucket-Richtlinie für einen Amazon-S3-on-Outposts-Bucket.

Themen

- Verwalten des Zugriffs auf einen Amazon S3-on-Outposts-Bucket basierend auf bestimmten IP-Adressen

Verwalten des Zugriffs auf einen Amazon S3-on-Outposts-Bucket basierend auf bestimmten IP-Adressen

Eine Bucket-Richtlinie ist eine auf Ressourcen basierende AWS Identity and Access Management-(IAM)-Richtlinie, die Sie verwenden können, um Zugriffsberechtigungen für Ihren Bucket und die darin enthaltenen Objekte zu erteilen. Nur der Bucket-Eigentümer kann einem Bucket eine Richtlinie zuordnen. Die dem Bucket zugeordneten Berechtigungen gelten für alle Objekte im Bucket, die dem Bucket-Eigentümer gehören. Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt. Weitere Informationen finden Sie unter Bucket-Richtlinie.

Beschränken des Zugriffs auf bestimmte IP-Adressen

Das folgende Beispiel verweigert allen Benutzern das Ausführen von S3-on-Outposts-Operationen für Objekte in den angegebenen Buckets, es sei denn, die Anforderung stammt aus dem angegebenen IP-Adressbereich.

> **Note**
>
> Wenn Sie den Zugriff auf eine bestimmte IP-Adresse einschränken, stellen Sie sicher, dass Sie auch angeben, welche VPC-Endpunkte, VPC-Quell-IP-Adressen oder externen IP-Adressen auf den S3-on-Outposts-Bucket zugreifen können. Andernfalls verlieren Sie möglicherweise den Zugriff auf den Bucket, wenn Ihre Richtlinie allen Benutzern die Ausführung von s3outposts Vorgängen an Objekten in Ihrem S3-on-Outposts-Bucket verweigert, ohne dass bereits die entsprechenden Berechtigungen vorhanden sind.

Die `Condition` Anweisung dieser Richtlinie identifiziert *`192.0.2.0/24`* als Bereich zulässiger IP-Adressen der Version 4 (IPv4).

Der `Condition`-Block verwendet die Bedingungen `NotIpAddress` und den Bedingungsschlüssel `aws:SourceIp`, wobei es sich um einen AWS-übergreifenden Bedingungsschlüssel handelt. Der `aws:SourceIp`-Bedingungsschlüssel kann nur für öffentliche IP-Adressbereiche verwendet werden. Weitere Informationen zu diesen Bedingungsschlüsseln finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für S3 on Outposts. Die `aws:SourceIp`-IPv4-Werte verwenden die CIDR-Standardnotation. Weitere Informationen finden Sie in der Referenz zu IAM-JSON-Richtlinienelementen im IAM-Benutzerhandbuch.

> **Warning**
>
> Bevor Sie diese S3-on-Outposts-Richtlinie verwenden, ersetzen Sie den *192.0.2.0/24* IP-Adressbereich in diesem Beispiel durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls verlieren Sie die Möglichkeit, auf Ihren Bucket zuzugreifen.

```
{
    "Version": "2012-10-17",
    "Id": "S3OutpostsPolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3outposts:*",
            "Resource": [
                "arn:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
accesspoint/EXAMPLE-ACCESS-POINT-NAME"
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/DOC-EXAMPLE-BUCKET"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                }
            }
        }
    ]
}
```

Zulassen von IPv4- und IPv6-Adressen

Wenn Sie mit der Verwendung von IPv6-Adressen beginnen, empfehlen wir, dass Sie alle Richtlinien Ihrer Organisation zusätzlich zu Ihren bereits vorhandenen IPv4-Adressbereichen auf Ihre IPv6-Adressbereiche aktualisieren. Auf diese Weise können Sie sicherstellen, dass die Richtlinien während der Umstellung auf IPv6 weiterhin funktionieren.

Die folgende Bucket-Beispielrichtlinie für S3 on Outposts zeigt, wie Sie IPv4- und IPv6-Adressbereiche kombinieren, um alle gültigen IP-Adressen Ihrer Organisation abzudecken. Die

Beispielrichtlinie erteilt Zugriff auf die IP-Adressen *192.0.2.1* und *2001:DB8:1234:5678::1* und verweigert den Zugriff auf die Adressen *203.0.113.1* und *2001:DB8:1234:5678:ABCD::1*.

Der `aws:SourceIp`-Bedingungsschlüssel kann nur für öffentliche IP-Adressbereiche verwendet werden. Die IPv6-Werte für `aws:SourceIp` müssen im CIDR-Standardformat angegeben werden. Für IPv6 unterstützen wir die Verwendung von `::` zur Darstellung eines Bereichs von Nullen (z. B. `2001:DB8:1234:5678::/64`). Weitere Informationen finden Sie unter IP-Adressen-Bedingungsoperatoren im IAM-Benutzerhandbuch.

> **Warning**
>
> Ersetzen Sie die IP-Adressbereiche in diesem Beispiel durch die entsprechenden Werte für Ihren Anwendungsfall, bevor Sie diese S3-on-Outposts-Richtlinie verwenden. Andernfalls verlieren Sie möglicherweise die Möglichkeit, auf Ihren Bucket zuzugreifen.

```
{
    "Id": "S3OutpostsPolicyId2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIPmix",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3outposts:*",
            "Resource": [
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/DOC-EXAMPLE-BUCKET",
                        "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-
ID/bucket/DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                },
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
```

```
                    "2001:DB8:1234:5678:ABCD::/80"
                ]
            }
        }
    }
  ]
}
```

Auflisten von Amazon-S3-on-Outposts-Buckets

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (OUTPOSTS). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Weitere Informationen über das Arbeiten mit Buckets in S3 on Outposts finden Sie unter Arbeiten mit S3-on-Outposts-Buckets.

Die folgenden Beispiele veranschaulichen, wie Sie eine Liste Ihrer S3-on-Outposts-Buckets mit der AWS Management Console, der AWS CLI und AWS SDK for Java zurückgeben.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Sehen Sie sich Ihre Liste der S3-on-Outposts-Buckets unter Outposts buckets (Outposts-Buckets) an.

Verwendung der AWS CLI

Im folgenden AWS CLI-Beispiel wird eine Liste von Buckets in einem Outpost abgerufen. Zum Verwenden dieses Befehls ersetzen Sie *user input placeholder* durch Ihre eigenen

Informationen. Weitere Informationen zu diesem Befehl finden Sie unter list-regional-buckets in der AWS CLI-Referenz.

```
aws s3control list-regional-buckets --account-id 123456789012 --outpost-
id op-01ac5d28a6a232904
```

Verwenden des AWS-SDKs für Java

Im folgenden SDK für Java-Beispiel wird eine Liste von Buckets in einem Outpost abgerufen. Weitere Informationen finden Sie unter ListRegionalBuckets in der API-Referenz zu Amazon Simple Storage Service.

```
import com.amazonaws.services.s3control.model.*;

public void listRegionalBuckets() {

    ListRegionalBucketsRequest reqListBuckets = new ListRegionalBucketsRequest()
            .withAccountId(AccountId)
            .withOutpostId(OutpostId);

    ListRegionalBucketsResult respListBuckets =
 s3ControlClient.listRegionalBuckets(reqListBuckets);
    System.out.printf("ListRegionalBuckets Response: %s%n",
 respListBuckets.toString());

}
```

Abrufen eines S3-on-Outposts-Buckets mithilfe der AWS CLI und des SDK for Java

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (OUTPOSTS). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS

Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Die folgenden Beispiele veranschaulichen, wie Sie mithilfe der AWS CLI und AWS SDK for Java einen S3-on-Outposts-Bucket abrufen.

> **Note**
>
> Wenn Sie mit Amazon S3 on Outposts über die AWS CLI oder AWS-SDKs arbeiten, geben Sie den Zugriffspunkt-ARN für den Outpost anstelle des Bucket-Namens an. Der Zugriffspunkt-ARN nimmt das folgende Format an, wobei *region* der AWS-Region-Code für die Region ist, in der sich der Outpost befindet:
> `arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point`
> Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Verwendung der AWS CLI

Im folgenden Beispiel wird ein S3-on-Outposts-Bucket mit der AWS CLI abgerufen. Zum Verwenden dieses Befehls ersetzen Sie *user input placeholder* durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter get-bucket in der AWS CLI-Referenz.

```
aws s3control get-bucket --account-id 123456789012 --bucket "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket"
```

Verwenden des AWS-SDKs für Java

Im folgenden Beispiel für S3 on Outposts wird ein Bucket mit dem SDK for Java abgerufen. Weitere Informationen finden Sie unter GetBucket in der API-Referenz zu Amazon Simple Storage Service.

```
import com.amazonaws.services.s3control.model.*;

public void getBucket(String bucketArn) {

    GetBucketRequest reqGetBucket = new GetBucketRequest()
            .withBucket(bucketArn)
```

```
            .withAccountId(AccountId);

    GetBucketResult respGetBucket = s3ControlClient.getBucket(reqGetBucket);
    System.out.printf("GetBucket Response: %s%n", respGetBucket.toString());

}
```

Löschen Ihres Amazon-S3-on-Outposts-Buckets

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (OUTPOSTS). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Weitere Informationen über das Arbeiten mit Buckets in S3 on Outposts finden Sie unter Arbeiten mit S3-on-Outposts-Buckets.

Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das ihn löschen kann.

> **Note**
>
> - Outposts-Buckets müssen leer sein, bevor sie gelöscht werden können.
>
> Die Amazon-S3-Konsole unterstützt keine S3-on-Outposts-Objektaktionen. Wenn Sie Objekte in Ihren S3-on-Outposts-Bucket hochladen und verwalten möchten, können Sie die REST-API, die AWS CLI oder AWS-SDKs verwenden.
> - Bevor Sie einen Outposts-Bucket löschen können, müssen Sie alle Outposts-Zugriffspunkte für den Bucket löschen. Weitere Informationen finden Sie unter Löschen eines Zugriffspunkts.
> - Sie können einen Bucket nicht wiederherstellen, nachdem er gelöscht wurde.

Die folgenden Beispiele veranschaulichen, wie Sie einen S3-on-Outposts-Bucket mithilfe der AWS Management Console und der AWS Command Line Interface (AWS CLI) löschen.

Verwenden der S3-Konsole

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Bucket, den Sie löschen möchten, und wählen Sie Delete (Löschen).
4. Bestätigen Sie das Löschen.

Verwendung der AWS CLI

Im folgenden Beispiel wird ein S3-on-Outposts-Bucket (`s3-outposts:DeleteBucket`) mithilfe der AWS CLI gelöscht. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control delete-bucket --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

Arbeiten mit Zugriffspunkten von Amazon S3 on Outposts

Um auf Ihren Amazon-S3-on-Outposts-Bucket zuzugreifen, müssen Sie einen Zugriffspunkt erstellen und konfigurieren.

Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie Amazon S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Zugriffspunkte unterstützen ausschließlich die virtuelle, Host-artige Adressierung.

> **Note**
>
> Das AWS-Konto, das den Outposts-Bucket erstellt, besitzt ihn und ist das einzige, das ihm Zugriffspunkte zuweisen kann.

In den folgenden Abschnitten wird beschrieben, wie Sie die Zugriffspunkte für S3-on-Outposts-Buckets erstellen und verwalten.

Themen

Erstellen eines S3-on-Outposts-Zugriffspunkten

Um auf Ihren Amazon-S3-on-Outposts-Bucket zuzugreifen, müssen Sie einen Zugriffspunkt erstellen und konfigurieren.

Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie Amazon S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Zugriffspunkte unterstützen ausschließlich die virtuelle, Host-artige Adressierung.

In den folgenden Beispielen wird das Erstellen eines Zugriffspunkts für S3 on Outposts mithilfe der AWS Management Console, der AWS Command Line Interface (AWS CLI) und AWS SDK for Java veranschaulicht.

> **Note**
>
> Das AWS-Konto, das den Outposts-Bucket erstellt, besitzt ihn und ist das einzige, das ihm Zugriffspunkte zuweisen kann.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, für den Sie einen Outposts-Zugriffspunkt erstellen möchten.
4. Wählen Sie die Registerkarte Outposts-Zugriffspunkte.
5. Wählen Sie im Abschnitt Outposts-Zugriffspunkte die Option Outposts-Zugriffspunkte erstellen aus.
6. Geben Sie im Abschnitt Outposts access point settings (Einstellungen für den Outposts-Zugriffspunkt) einen Namen für den Zugriffspunkt ein und wählen Sie die Virtual Private Cloud (VPC) für den Zugriffspunkt aus.
7. Wenn Sie eine Richtlinie für Ihren Zugriffspunkt hinzufügen möchten, geben Sie sie in den Abschnitt Richtlinien für den Outposts-Zugriffspunkt ein.

   Weitere Informationen finden Sie unter Einrichten von IAM mit S3 on Outposts.

Verwendung der AWS CLI

Example

Im folgenden AWS CLI-Beispiel wird ein Zugriffspunkt für einen Outposts-Bucket erstellt. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

Verwenden des AWS-SDKs für Java

Example

Im folgenden Beispiel für SDK für Java wird ein Zugriffspunkt für einen Outposts-Bucket erstellt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
import com.amazonaws.services.s3control.model.*;

public String createAccessPoint(String bucketArn, String accessPointName) {
```

```
    CreateAccessPointRequest reqCreateAP = new CreateAccessPointRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withName(accessPointName)
            .withVpcConfiguration(new VpcConfiguration().withVpcId("vpc-12345"));

    CreateAccessPointResult respCreateAP =
 s3ControlClient.createAccessPoint(reqCreateAP);
    System.out.printf("CreateAccessPoint Response: %s%n", respCreateAP.toString());

    return respCreateAP.getAccessPointArn();

}
```

Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets

Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Jedes Mal, wenn Sie einen Zugriffspunkt für einen Bucket erstellen, generiert S3 on Outposts automatisch einen Zugriffspunkt-Alias. Sie können diesen Zugriffspunkt-Alias anstelle eines Zugriffspunkt-ARNs für jede Datenebenen-Operation verwenden. Sie können beispielsweise einen Zugriffspunkt-Alias verwenden, um Operationen auf Objektebene wie PUT, GET, LIST und mehr auszuführen. Eine Liste dieser Vorgänge finden Sie unter Amazon-S3-API-Vorgänge für die Objektverwaltung.

Das folgende Beispiel zeigt einen ARN- und Zugriffspunkt-Alias für einen Zugriffspunkt namens *my-access-point*.

- Zugriffspunkt-ARN – `arn:aws:s3-outposts:`*region*`:`*123456789012*`:outpost/`*op-01ac5d28a6a232904*`/accesspoint/`*my-access-point*
- Zugriffspunkt-Alias – *my-access-po-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10*`--op-s3`

Weitere Informationen zur Verwendung von ARNs finden Sie unter Amazon-Ressourcennamen (ARN) im Allgemeine AWS-Referenz.

Weitere Informationen über die Zugriffspunkt-Aliasse finden Sie in den folgenden Themen.

Themen

- Zugriffspunkt-Aliasse
- Verwenden eines Zugriffspunkt-Alias in einer Objektoperation von S3 on Outposts
- Einschränkungen

Zugriffspunkt-Aliasse

Ein Zugriffspunkt-Alias wird innerhalb desselben Namespace wie ein S3-on-Outposts-Bucket erstellt. Wenn Sie einen Zugriffspunkt erstellen, generiert S3 on Outposts automatisch einen Zugriffspunkt-Alias, der nicht geändert werden kann. Ein Zugriffspunkt-Alias erfüllt alle Anforderungen eines gültigen Bucket-Namens von S3 on Outposts und besteht aus den folgenden Teilen:

access point name prefix`-`*metadata*`--op-s3`

> **Note**
>
> Das Suffix `--op-s3` ist für Zugriffspunkt-Aliasse reserviert. Wir empfehlen, es nicht für Bucket- oder Zugriffspunktnamen zu verwenden. Weitere Informationen zu Bucket-Benennungsregeln für S3 on Outposts finden Sie unter Arbeiten mit S3-on-Outposts-Buckets.

Suchen des Zugriffspunkt-Alias

Die folgenden Beispiele zeigen Ihnen, wie Sie einen Zugriffspunkt-Alias mit der Amazon-S3-Konsole und der AWS CLI finden.

Example : Suchen und Kopieren eines Zugriffspunkt-Alias in der Amazon-S3-Konsole

Nachdem Sie einen Zugriffspunkt in der Konsole erstellt haben, können Sie den Zugriffspunkt-Alias der Spalte Access Point alias (Zugriffspunkt-Alias) der Liste Access Points (Zugriffspunkte) entnehmen.

So kopieren Sie einen Zugriffspunkt-Alias

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts access points (Outposts--Zugriffspunkte) aus.
3. Zum Kopieren des Zugriffspunkt-Alias führen Sie einen der folgenden Schritte aus:
   - Wählen Sie in der Liste Access Points (Zugriffspunkte) das Optionsfeld neben dem Namen des Zugriffspunkts und dann Copy Access Point alias (Zugriffspunkt-Alias kopieren) aus.

- Wählen Sie den Namen des Zugriffspunkts aus. Kopieren Sie dann unter Outposts access point overview (Outposts-Zugriffspunkt – Übersicht) den Zugriffspunkt-Alias.

Example : Erstellen eines Zugriffspunkts mit der AWS CLI und Suchen des Zugriffspunkt-Alias in der Antwort

Im folgenden AWS CLI-Beispiel für den `create-access-point`-Befehl wird der Zugriffspunkt erstellt und der automatisch generierte Zugriffspunkt-Alias zurückgegeben. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control create-access-point --bucket example-outposts-bucket --name example-
outposts-access-point --account-id 123456789012

{
    "AccessPointArn":
    "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
accesspoint/example-outposts-access-point",
    "Alias": "example-outp-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10--op-s3"
}
```

Example : Abrufen eines Zugriffspunkt-Alias mithilfe der AWS CLI

Das folgende AWS CLI-Beispiel für den `get-access-point`-Befehl gibt Informationen über den angegebenen Zugriffspunkt zurück. Diese Informationen enthalten den Zugriffspunkt-Alias. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control get-access-point --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --name example-outposts-access-point --account-id 123456789012

{
    "Name": "example-outposts-access-point",
    "Bucket": "example-outposts-bucket",
    "NetworkOrigin": "Vpc",
    "VpcConfiguration": {
        "VpcId": "vpc-01234567890abcdef"
    },
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": true,
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
```

```
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2022-09-18T17:49:15.584000+00:00",
    "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
}
```

Example : Auflisten der Zugriffspunkte, um einen Zugriffspunkt-Alias mithilfe der AWS CLI zu finden

Das folgende AWS CLI-Beispiel für den `list-access-points`-Befehl listet Informationen über den angegebenen Zugriffspunkt auf. Diese Informationen enthalten den Zugriffspunkt-Alias. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket

{
    "AccessPointList": [
        {
            "Name": "example-outposts-access-point",
            "NetworkOrigin": "Vpc",
            "VpcConfiguration": {
                "VpcId": "vpc-01234567890abcdef"
            },
            "Bucket": "example-outposts-bucket",
            "AccessPointArn": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point",
            "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
        }
    ]
}
```

Verwenden eines Zugriffspunkt-Alias in einer Objektoperation von S3 on Outposts

Bei der Übernahme von Zugriffspunkten können Sie Zugriffspunkt-Aliasse verwenden, ohne dass umfangreiche Codeänderungen erforderlich sind.

Dieses AWS CLI-Beispiel zeigt eine `get-object`-Operation für einen Bucket von S3 on Outposts. In diesem Beispiel wird anstelle des vollständigen Zugriffspunkt-ARN der Zugriffspunkt-Alias als Wert für `--bucket` verwendet.

```
aws s3api get-object --bucket my-access-po-
o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3 --key  testkey  sample-object.rtf

{
    "AcceptRanges": "bytes",
    "LastModified": "2020-01-08T22:16:28+00:00",
    "ContentLength": 910,
    "ETag": "\"00751974dc146b76404bb7290f8f51bb\"",
    "VersionId": "null",
    "ContentType": "text/rtf",
    "Metadata": {}
}
```

Einschränkungen

- Aliase können nicht von Kunden konfiguriert werden.
- Aliasse können auf einem Zugriffspunkt nicht gelöscht, geändert oder deaktiviert werden.
- Sie können einen Zugriffspunkt-Alias nicht für Kontrollebenen-Operationen von S3 on Outposts verwenden. Eine Liste von Steuerebenen-Operationen von S3 on Outposts finden Sie unter Amazon-S3-Control-API-Vorgänge zum Verwalten von Buckets.
- Aliasse können in AWS Identity and Access Management (IAM)-Richtlinien nicht verwendet werden.

Anzeigen von Informationen über eine Zugriffspunktkonfiguration

Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie Amazon S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Zugriffspunkte unterstützen ausschließlich die virtuelle, Host-artige Adressierung.

In den folgenden Themen erfahren Sie, wie Sie Konfigurationsinformationen für einen S3-on-Outposts-Zugriffspunkt mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) und AWS SDK for Java zurückgeben können.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie im Navigationsbereich Outposts access points (Outposts--Zugriffspunkte) aus.
3. Wählen Sie den Outposts-Zugriffspunkt aus, für den Sie Konfigurationsdetails anzeigen möchten.
4. Sehen Sie sich unter Outposts access point overview (Übersicht über Outposts-Zugriffspunkte) die Konfigurationsdetails zum Zugriffspunkt an.

Verwendung der AWS CLI

Das folgende AWS CLI-Beispiel ruft einen Zugriffspunkt für einen Outposts-Bucket ab. Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control get-access-point --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

Verwenden des AWS-SDKs für Java

Im folgenden Beispiel für SDK für Java wird ein Zugriffspunkt für einen Outposts-Bucket abgerufen.

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPoint(String accessPointArn) {

    GetAccessPointRequest reqGetAP = new GetAccessPointRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointResult respGetAP = s3ControlClient.getAccessPoint(reqGetAP);
    System.out.printf("GetAccessPoint Response: %s%n", respGetAP.toString());

}
```

Eine Liste Ihrer Amazon-S3-on-Outposts-Zugriffspunkte anzeigen

Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie Amazon S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Zugriffspunkte unterstützen ausschließlich die virtuelle, Host-artige Adressierung.

Die folgenden Themen zeigen Ihnen, wie Sie eine Liste Ihrer S3-on-Outposts-Zugriffspunkte mit der AWS Management Console, AWS Command Line Interface (AWS CLI) und AWS SDK for Java auflisten.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts access points (Outposts--Zugriffspunkte) aus.
3. Sehen Sie sich Ihre Liste der S3-on-Outposts-Zugriffspunkte unter Outposts access points(Outposts-Zugriffspunkte) an.

Verwendung der AWS CLI

Das folgende AWS CLI-Beispiel listet Zugriffspunkte für einen Outposts-Bucket auf. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

Verwenden des AWS-SDKs für Java

Im folgenden Beispiel für SDK für Java werden Zugriffspunkte für einen Outposts-Bucket aufgelistet.

```
import com.amazonaws.services.s3control.model.*;

public void listAccessPoints(String bucketArn) {

    ListAccessPointsRequest reqListAPs = new ListAccessPointsRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    ListAccessPointsResult respListAPs = s3ControlClient.listAccessPoints(reqListAPs);
    System.out.printf("ListAccessPoints Response: %s%n", respListAPs.toString());

}
```

Löschen eines Zugriffspunkts

Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in Amazon S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie Amazon S3-Objekt-Vorgänge ausführen können, z. B. `GetObject` und `PutObject`. Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Zugriffspunkte unterstützen ausschließlich die virtuelle, Host-artige Adressierung.

Die folgenden Beispiele zeigen Ihnen, wie Sie einen Zugriffspunkt mit der AWS Management Console und der AWS Command Line Interface (AWS CLI) löschen.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts access points (Outposts-Zugriffspunkte) aus.
3. Wählen Sie im Bereich Outposts-Zugriffspunkte den Outposts-Zugriffspunkt aus, den Sie löschen möchten.
4. Wählen Sie Delete (Löschen).
5. Bestätigen Sie das Löschen.

Verwendung der AWS CLI

Im folgenden AWS CLI-Beispiel wird ein Outposts-Zugriffspunkt gelöscht. Zum Ausführen dieses Befehls ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control delete-access-point --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

Hinzufügen oder Bearbeiten einer Zugriffspunktrichtlinie

Jeder Zugriffspunkt verfügt über unterschiedliche Berechtigungen und Netzwerkkontrollen, die Amazon S3 on Outposts auf alle Anforderungen anwendet, die über diesen Zugriffspunkt eingehen. Jeder Zugriffspunkt erzwingt eine benutzerdefinierte Zugriffspunktrichtlinie, die in Verbindung mit der Bucket-Richtlinie funktioniert, die dem zugrunde liegenden Bucket zugeordnet ist. Weitere Informationen finden Sie unter Zugriffspunkte.

Die folgenden Themen zeigen Ihnen, wie Sie die Zugriffspunktsrichtlinie für S3 on Outposts mithilfe von AWS Management Console, AWS Command Line Interface (AWS CLI) und AWS SDK for Java hinzufügen oder bearbeiten.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, für den Sie die Zugriffspunktsrichtlinie bearbeiten möchten.
4. Wählen Sie die Registerkarte Outposts-Zugriffspunkte.
5. Wählen Sie im Abschnitt Outposts-Zugriffspunkte den Zugriffspunkt aus, dessen Richtlinie Sie bearbeiten möchten, und wählen Sie Richtlinie bearbeiten.
6. Fügen Sie die Richtlinie im Abschnitt Richtlinien für den Outposts-Zugriffspunkt hinzu oder bearbeiten Sie sie. Weitere Informationen finden Sie unter Einrichten von IAM mit S3 on Outposts.

Verwendung der AWS CLI

Im folgenden AWS CLI-Beispiel wird eine Richtlinie für einen Outposts-Zugriffspunkt eingerichtet.

1. Speichern Sie die folgende Zugriffspunktrichtlinie in einer JSON-Datei. In diesem Beispiel heißt die Datei `appolicy1.json`. Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
{
   "Version":"2012-10-17",
   "Id":"exampleAccessPointPolicy",
   "Statement":[
      {
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point
```

```
      }
   ]
}
```

2. Senden Sie die JSON-Datei als Teil des CLI-Befehls `put-access-point-policy`. Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
aws s3control put-access-point-policy --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --policy file://appolicy1.json
```

Verwenden des AWS-SDKs für Java

Im folgenden SDK-für-Java-Beispiel wird eine Richtlinie für einen Outposts-Zugriffspunkt eingerichtet.

```
import com.amazonaws.services.s3control.model.*;

public void putAccessPointPolicy(String accessPointArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testAccessPointPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId + "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + accessPointArn +
 "\"}]}";

    PutAccessPointPolicyRequest reqPutAccessPointPolicy = new
 PutAccessPointPolicyRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn)
            .withPolicy(policy);

    PutAccessPointPolicyResult respPutAccessPointPolicy =
 s3ControlClient.putAccessPointPolicy(reqPutAccessPointPolicy);
    System.out.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());
    printWriter.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());

}
```

Anzeigen einer Zugriffspunktsrichtlinie für einen S3-on-Outposts-Zugriffspunkt

Jeder Zugriffspunkt verfügt über unterschiedliche Berechtigungen und Netzwerkkontrollen, die Amazon S3 on Outposts auf alle Anforderungen anwendet, die über diesen Zugriffspunkt eingehen. Jeder Zugriffspunkt erzwingt eine benutzerdefinierte Zugriffspunktrichtlinie, die in Verbindung mit der Bucket-Richtlinie funktioniert, die dem zugrunde liegenden Bucket zugeordnet ist. Weitere Informationen finden Sie unter Zugriffspunkte.

Weitere Informationen zum Arbeiten mit Zugriffspunkten in S3 on Outposts finden Sie unter Arbeiten mit S3-on-Outposts-Buckets.

Die folgenden Themen zeigen Ihnen, wie Sie die Zugriffspunktrichtlinie für S3 on Outposts mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) und AWS SDK for Java anzeigen.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts access points (Outposts--Zugriffspunkte) aus.
3. Wählen Sie den Outposts-Zugriffspunkt aus, für den Sie die Richtlinie anzeigen möchten.
4. Überprüfen Sie auf dem Tab Permissions (Berechtigungen) die Zugriffspunktrichtlinie für S3 on Outposts.
5. Weitere Informationen zum Bearbeiten der Zugriffspunktrichtlinie finden Sie unterHinzufügen oder Bearbeiten einer Zugriffspunktrichtlinie.

Verwendung der AWS CLI

Im folgenden AWS CLI-Beispiel wird eine Richtlinie für einen Outposts-Zugriffspunkt abgerufen. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3control get-access-point-policy --account-id 123456789012 --name arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point
```

Verwenden des AWS-SDKs für Java

Im folgenden SDK-für-Java-Beispiel wird eine Richtlinie für einen Outposts-Zugriffspunkt abgerufen.

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPointPolicy(String accessPointArn) {

    GetAccessPointPolicyRequest reqGetAccessPointPolicy = new
 GetAccessPointPolicyRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointPolicyResult respGetAccessPointPolicy =
 s3ControlClient.getAccessPointPolicy(reqGetAccessPointPolicy);
    System.out.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());
    printWriter.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());

}
```

Arbeiten mit Amazon-S3-on-Outposts-Endpunkten

Wenn Sie Anforderungen an einen Zugriffspunkt für Amazon S3 on Outposts weiterleiten möchten, müssen Sie einen S3-on-Outposts-Endpunkt erstellen und konfigurieren. Zum Erstellen eines Endpunkts muss Ihr Service-Link eine aktive Verbindung mit Ihrer Outposts-Heimatregion aufweisen. Jeder Virtual Private Cloud (VPC) in Ihrem Outpost kann ein Endpunkt zugeordnet sein. Weitere Informationen zu Endpunktkontingenten finden Sie unter Netzwerkanforderungen von S3 on Outposts. Sie müssen einen Endpunkt erstellen, um auf Ihre Outposts-Buckets zugreifen und Objekt-Operationen ausführen zu können. Weitere Informationen finden Sie unter Endpunkte.

Nachdem Sie einen Endpunkt erstellt haben, finden Sie im Feld „Status“ weitere Informationen zum Status des Endpunkts. Wenn Ihre Outposts offline sind, wird CREATE_FAILED zurückgegeben. Sie können Ihre Service-Link-Verbindung überprüfen, den Endpunkt löschen und das Erstellen erneut versuchen, nachdem die Verbindung wiederhergestellt wurde. Eine Liste mit zusätzlichen Fehlercodes finden Sie nachstehend. Weitere Informationen finden Sie unter Endpunkte.

API	Status	Grund für Fehlschlag – Fehlercode	Meldung – Grund für Fehlschlag
`CreateEndpoint`	Create_Failed	OutpostNotReachable	Der Endpunkt konnte nicht erstellt werden, da die Service-Link-Verbindung zur Outposts-Heimatregion unterbroc

API	Status	Grund für Fehlschlag – Fehlercode	Meldung – Grund für Fehlschlag
			hen ist. Überprüfen Sie Ihre Verbindun g, löschen Sie den Endpunkt und versuchen Sie es erneut.
`CreateEndpoint`	Create_Failed	InternalError	Der Endpunkt konnte aufgrund eines internen Fehlers nicht erstellt werden. Bitte löschen Sie den Endpunkt und erstellen Sie ihn erneut.
`DeleteEndpoint`	Delete_Failed	OutpostNotReachable	Der Endpunkt konnte nicht gelöscht werden, da die Service-Link-Verbindung zur Outposts-Heimatregion unterbroc hen ist. Überprüfen Sie Ihre Verbindung und versuchen Sie es erneut.
`DeleteEndpoint`	Delete_Failed	InternalError	Der Endpunkt konnte aufgrund eines internen Fehlers nicht gelöscht werden. Bitte versuchen Sie es noch einmal.

Weitere Informationen über das Arbeiten mit Buckets in S3 on Outposts finden Sie unter Arbeiten mit S3-on-Outposts-Buckets.

In den folgenden Abschnitten wird die Erstellung und Verwaltung von Endpunkten für S3 on Outposts beschrieben.

Themen

- Erstellen eines Endpunkts in einem Outpost
- Anzeigen einer Liste Ihrer Amazon-S3-on-Outposts-Endpunkte
- Löschen eines Amazon-S3-on-Outposts-Endpunkts

Erstellen eines Endpunkts in einem Outpost

Wenn Sie Anforderungen an einen Zugriffspunkt für Amazon S3 on Outposts weiterleiten möchten, müssen Sie einen S3-on-Outposts-Endpunkt erstellen und konfigurieren. Zum Erstellen eines

Endpunkts muss Ihr Service-Link eine aktive Verbindung mit Ihrer Outposts-Heimatregion aufweisen. Jeder Virtual Private Cloud (VPC) in Ihrem Outpost kann ein Endpunkt zugeordnet sein. Weitere Informationen zu Endpunktkontingenten finden Sie unter Netzwerkanforderungen von S3 on Outposts. Sie müssen einen Endpunkt erstellen, um auf Ihre Outposts-Buckets zugreifen und Objekt-Operationen ausführen zu können. Weitere Informationen finden Sie unter Endpunkte.

Berechtigungen

Weitere Informationen zu den erforderlichen Berechtigungen für das Erstellen eines Endpunkts finden Sie unter Berechtigungen für S3-on-Outposts-Endpunkte.

Wenn Sie einen Endpunkt erstellen, erstellt S3 on Outposts auch eine serviceverknüpfte Rolle in Ihrem AWS-Konto. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für S3 on Outposts.

Die folgenden Beispiele zeigen, wie Sie einen S3-on-Outposts-Endpunkt mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) und AWS SDK for Java erstellen.

Verwenden der S3-Konsole

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts access points (Outposts-Zugriffspunkte) aus.
3. Wählen Sie den Tab Outposts endpoints (Outposts-Endpunkte) aus.
4. Wählen Sie Create Outposts endpoint (Outposts-Endpunkt erstellen) aus.
5. Wählen Sie unter Outpost den Outpost aus, auf dem dieser Endpunkt erstellt werden soll.
6. Wählen Sie unter VPC eine VPC aus, die noch keinen Endpunkt hat und außerdem den Regeln für Outposts-Endpunkte entspricht.

   Eine Virtual Private Cloud (VPC) ermöglicht es Ihnen, AWS-Ressourcen in einem virtuellen Netzwerk zu launchen, das Sie definieren. Dieses virtuelle Netzwerk entspricht weitgehend einem herkömmlichen Netzwerk, wie Sie es in Ihrem Rechenzentrum betreiben würden, kann jedoch die Vorzüge der skalierbaren Infrastruktur von nutze AWS.

   Wenn Sie keine VPC haben, wählen Sie VPC erstellen. Weitere Informationen finden Sie unter Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind.
7. Wählen Sie Create Outposts endpoint (Outposts-Endpunkt erstellen) aus.

Verwendung von AWS CLI

Example

Im folgenden AWS CLI-Beispiel wird ein Endpunkt für einen Outpost mithilfe des VPC-Ressourcenzugriffstyps erstellt. Die VPC ist vom Subnetz abgeleitet. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

Im folgenden AWS CLI-Beispiel wird ein Endpunkt für einen Outpost mit dem Zugriffstyp des kundeneigenen IP-Adresspools (CoIP-Pool) erstellt. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

Verwenden des AWS-SDKs für Java

Example

Im folgenden SDK für Java-Beispiel wird ein Endpunkt für einen Outpost erstellt. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.CreateEndpointRequest;
import com.amazonaws.services.s3outposts.model.CreateEndpointResult;

public void createEndpoint() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    CreateEndpointRequest createEndpointRequest = new CreateEndpointRequest()
                .withOutpostId("op-0d79779cef3c30a40")
                .withSubnetId("subnet-8c7a57c5")
                .withSecurityGroupId("sg-ab19e0d1")
                .withAccessType("CustomerOwnedIp")
                .withCustomerOwnedIpv4Pool("ipv4pool-coip-12345678901234567");
```

```
    // Use .withAccessType and .withCustomerOwnedIpv4Pool only when the access type is
    // customer-owned IP address pool (CoIP pool)
    CreateEndpointResult createEndpointResult =
 s3OutpostsClient.createEndpoint(createEndpointRequest);
    System.out.println("Endpoint is created and its ARN is " +
 createEndpointResult.getEndpointArn());
}
```

Anzeigen einer Liste Ihrer Amazon-S3-on-Outposts-Endpunkte

Wenn Sie Anforderungen an einen Zugriffspunkt für Amazon S3 on Outposts weiterleiten möchten, müssen Sie einen S3-on-Outposts-Endpunkt erstellen und konfigurieren. Zum Erstellen eines Endpunkts muss Ihr Service-Link eine aktive Verbindung mit Ihrer Outposts-Heimatregion aufweisen. Jeder Virtual Private Cloud (VPC) in Ihrem Outpost kann ein Endpunkt zugeordnet sein. Weitere Informationen zu Endpunktkontingenten finden Sie unter Netzwerkanforderungen von S3 on Outposts. Sie müssen einen Endpunkt erstellen, um auf Ihre Outposts-Buckets zugreifen und Objekt-Operationen ausführen zu können. Weitere Informationen finden Sie unter Endpunkte.

Die folgenden Beispiele veranschaulichen, wie Sie eine Liste Ihrer S3-on-Outposts-Endpunkte mit der AWS Management Console, der AWS Command Line Interface (AWS CLI) und AWS SDK for Java zurückgeben.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts access points (Outposts-Zugriffspunkte) aus.
3. Wählen Sie auf der Seite Outposts access points (Outposts-Zugriffspunkte) den Tab Outposts endpoints (Outposts-Endpunkte) aus.
4. Unter Outposts endpoints (Outposts-Endpunkte) können Sie eine Liste Ihrer S3-on-Outposts-Endpunkte anzeigen.

Verwendung von AWS CLI

Das folgende AWS CLI-Beispiel listet die Endpunkte für die AWS Outposts-Ressourcen auf, die Ihrem Konto zugeordnet sind. Weitere Informationen über diesen Befehl finden Sie unter list-endpoints in der AWS CLI-Referenz.

```
aws s3outposts list-endpoints
```

Verwenden des AWS-SDKs für Java

Im folgenden SDK für Java-Beispiel werden Endpunkte für einen Outpost aufgelistet. Weitere Informationen finden Sie unter ListEndpoints in der API-Referenz zu Amazon Simple Storage Service.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.ListEndpointsRequest;
import com.amazonaws.services.s3outposts.model.ListEndpointsResult;

public void listEndpoints() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    ListEndpointsRequest listEndpointsRequest = new ListEndpointsRequest();
    ListEndpointsResult listEndpointsResult =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
    System.out.println("List endpoints result is " + listEndpointsResult);
}
```

Löschen eines Amazon-S3-on-Outposts-Endpunkts

Wenn Sie Anforderungen an einen Zugriffspunkt für Amazon S3 on Outposts weiterleiten möchten, müssen Sie einen S3-on-Outposts-Endpunkt erstellen und konfigurieren. Zum Erstellen eines Endpunkts muss Ihr Service-Link eine aktive Verbindung mit Ihrer Outposts-Heimatregion aufweisen. Jeder Virtual Private Cloud (VPC) in Ihrem Outpost kann ein Endpunkt zugeordnet sein. Weitere Informationen zu Endpunktkontingenten finden Sie unter Netzwerkanforderungen von S3 on Outposts. Sie müssen einen Endpunkt erstellen, um auf Ihre Outposts-Buckets zugreifen und Objekt-Operationen ausführen zu können. Weitere Informationen finden Sie unter Endpunkte.

Die folgenden Beispiele veranschaulichen, wie Sie Ihre S3-on-Outposts-Endpunkte mit der AWS Management Console, der AWS Command Line Interface (AWS CLI) und AWS SDK for Java löschen.

Verwenden der S3-Konsole

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts access points (Outposts-Zugriffspunkte) aus.
3. Wählen Sie auf der Seite Outposts access points (Outposts-Zugriffspunkte) den Tab Outposts endpoints (Outposts-Endpunkte) aus.

4. Wählen Sie unter Outposts endpoints (Outposts-Endpunkte) den Endpunkt aus, den Sie löschen möchten, und klicken Sie dann auf Delete (Löschen).

Verwendung von AWS CLI

Im folgenden AWS CLI-Beispiel wird ein Endpunkt für einen Outpost gelöscht. Zum Ausführen dieses Befehls ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws s3outposts delete-endpoint --endpoint-id example-endpoint-id --outpost-
id op-01ac5d28a6a232904
```

Verwenden des AWS-SDKs für Java

Im folgenden SDK-für-Java-Beispiel wird ein Endpunkt für einen Outpost gelöscht. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
import com.amazonaws.arn.Arn;
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.DeleteEndpointRequest;

public void deleteEndpoint(String endpointArnInput) {
    String outpostId = "op-01ac5d28a6a232904";
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    Arn endpointArn = Arn.fromString(endpointArnInput);
    String[] resourceParts = endpointArn.getResource().getResource().split("/");
    String endpointId = resourceParts[resourceParts.length - 1];
    DeleteEndpointRequest deleteEndpointRequest = new DeleteEndpointRequest()
                .withEndpointId(endpointId)
                .withOutpostId(outpostId);
    s3OutpostsClient.deleteEndpoint(deleteEndpointRequest);
    System.out.println("Endpoint with id " + endpointId + " is deleted.");
}
```

Arbeiten mit S3-on-Outposts-Objekten

Mit Amazon S3 on Outposts können Sie S3-Buckets auf Ihren AWS Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale

Datenresidenz erfordern, einfach On-Premises speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse, S3 Outposts (OUTPOSTS), die die Amazon S3-APIs verwendet und darauf ausgelegt ist, Daten über mehrere Geräte und Server hinweg auf Ihrem dauerhaft und redundant zu speichern AWS Outposts. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDKs oder REST API verwenden.

Objekte sind die Grundeinheiten, die in Amazon S3 on Outposts gespeichert sind. Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Amazon-Ressourcennamen (ARN) oder den Alias des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Zugriffspunkte, das den AWS-Region Code für die Region, in der sich der Outpost befindet, die AWS-Konto ID, die Outpost-ID und den Namen des Zugriffspunkts enthält:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Objekt-ARNs verwenden das folgende Format, das die enthält AWS-Region , in der sich der Outpost befindet, AWS-Konto die ID, die Outpost-ID, den Bucket-Namen und den Objektschlüssel:

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/bucket/DOC-EXAMPLE-BUCKET1/object/myobject
```

Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn ein Outpost-Rack AWS installiert, bleiben Ihre Daten lokal in Ihrem Outpost, um die Anforderungen an die Datenresidenz zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da das in der Region gehostet AWS Management Console wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können jedoch die REST API, AWS Command Line Interface (AWS CLI) und AWS SDKs verwenden, um Ihre Objekte über Ihre Zugriffspunkte hochzuladen und zu verwalten.

Informationen zum Hochladen eines Objekts finden Sie unter Schritt 4: Hochladen eines Objekts in einen S3-on-Outposts-Bucket. Weitere Objektaktionen können Sie den folgenden Themen entnehmen.

Themen

- Kopieren eines Objekts in einem Amazon S3 on Outposts-Bucket mit AWS SDK for Java
- Abrufen eines Objekts aus einem Amazon-S3-on-Outposts-Bucket
- Auflisten der Objekten in einem Amazon-S3-on-Outposts-Bucket
- Löschen von Objekten in Amazon-S3-on.Outposts-Buckets
- Verwenden von HeadBucket, um festzustellen, ob ein S3-on-Outposts-Bucket vorhanden ist und Sie Zugriffsberechtigungen haben
- Durchführen und Verwalten eines mehrteiligen Uploads mit dem SDK for Java
- Verwenden vorsignierter URLs für S3 on Outposts
- Amazon S3 on Outposts mit lokalem Amazon EMR on Outposts
- Zwischenspeicherung von Autorisierung und Authentifizierung

Kopieren eines Objekts in einem Amazon S3 on Outposts-Bucket mit AWS SDK for Java

Objekte sind die Grundeinheiten, die in Amazon S3 on Outposts gespeichert sind. Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Amazon-Ressourcennamen (ARN) oder den Alias des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Objekte, das den AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, Outpost-ID, Bucket-Name und den Namen des Zugriffspunkts umfasst:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Das folgenden Beispiel veranschaulicht, wie Sie mithilfe von AWS SDK for Java ein Objekt in einem S3-on-Outposts-Bucket kopieren.

Verwenden des AWS-SDKs für Java

Im folgenden S3-on-Outposts-Beispiel wird ein Objekt mithilfe des SDK für Java in ein neues Objekt im selben Bucket kopiert. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;

public class CopyObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String sourceKey = "*** Source object key ***";
        String destinationKey = "*** Destination object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Copy the object into a new object in the same bucket.
            CopyObjectRequest copyObjectRequest = new CopyObjectRequest(accessPointArn,
 sourceKey, accessPointArn, destinationKey);
            s3Client.copyObject(copyObjectRequest);
        } catch (AmazonServiceException e) {
```

```
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Abrufen eines Objekts aus einem Amazon-S3-on-Outposts-Bucket

Objekte sind die Grundeinheiten, die in Amazon S3 on Outposts gespeichert sind. Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Amazon-Ressourcennamen (ARN) oder den Alias des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Objekte, das den AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, Outpost-ID, Bucket-Name und den Namen des Zugriffspunkts umfasst:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Die folgenden Beispiele veranschaulichen, wie Sie ein Objekt mithilfe der AWS Command Line Interface (AWS CLI) und AWS SDK for Java herunterladen (abrufen).

Verwendung der AWS CLI

Im folgenden Beispiel wird ein Objekt mit dem Namen `sample-object.xml` in einem S3-on-Outposts-Bucket (`s3-outposts:GetObject`) mit der AWS CLI abgerufen. Zum Verwenden dieses Befehls ersetzen Sie *`user input placeholder`* durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter get-object in der AWS CLI-Referenz.

```
aws s3api get-object --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point --key testkey sample-object.xml
```

Verwenden des AWS-SDKs für Java

Im folgenden Beispiel für S3 on Outposts wird ein Objekt mit dem SDK for Java abgerufen. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie jeden *`user input placeholder`* durch Ihre Informationen. Weitere Informationen finden Sie unter GetObject in der Amazon Simple Storage Service-API-Referenz.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GetObjectRequest;
import com.amazonaws.services.s3.model.ResponseHeaderOverrides;
import com.amazonaws.services.s3.model.S3Object;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

public class GetObject {
    public static void main(String[] args) throws IOException {
        String accessPointArn = "*** access point ARN ***";
        String key = "*** Object key ***";

        S3Object fullObject = null, objectPortion = null, headerOverrideObject = null;
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
```

```
                    .enableUseArnRegion()
                    .build();

            // Get an object and print its contents.
            System.out.println("Downloading an object");
            fullObject = s3Client.getObject(new GetObjectRequest(accessPointArn, key));
            System.out.println("Content-Type: " +
fullObject.getObjectMetadata().getContentType());
            System.out.println("Content: ");
            displayTextInputStream(fullObject.getObjectContent());

            // Get a range of bytes from an object and print the bytes.
            GetObjectRequest rangeObjectRequest = new GetObjectRequest(accessPointArn,
key)
                    .withRange(0, 9);
            objectPortion = s3Client.getObject(rangeObjectRequest);
            System.out.println("Printing bytes retrieved.");
            displayTextInputStream(objectPortion.getObjectContent());

            // Get an entire object, overriding the specified response headers, and
print the object's content.
            ResponseHeaderOverrides headerOverrides = new ResponseHeaderOverrides()
                    .withCacheControl("No-cache")
                    .withContentDisposition("attachment; filename=example.txt");
            GetObjectRequest getObjectRequestHeaderOverride = new
GetObjectRequest(accessPointArn, key)
                    .withResponseHeaders(headerOverrides);
            headerOverrideObject = s3Client.getObject(getObjectRequestHeaderOverride);
            displayTextInputStream(headerOverrideObject.getObjectContent());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        } finally {
            // To ensure that the network connection doesn't remain open, close any
open input streams.
            if (fullObject != null) {
                fullObject.close();
            }
            if (objectPortion != null) {
```

```
                objectPortion.close();
            }
            if (headerOverrideObject != null) {
                headerOverrideObject.close();
            }
        }
    }

    private static void displayTextInputStream(InputStream input) throws IOException {
        // Read the text input stream one line at a time and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
        String line = null;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

Auflisten der Objekten in einem Amazon-S3-on-Outposts-Bucket

Objekte sind die Grundeinheiten, die in Amazon S3 on Outposts gespeichert sind. Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Amazon-Ressourcennamen (ARN) oder den Alias des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Objekte, das den AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, Outpost-ID, Bucket-Name und den Namen des Zugriffspunkts umfasst:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

> **Note**
>
> Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen

an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Die folgenden Beispiele veranschaulichen, wie Sie mithilfe der AWS CLI und AWS SDK for Java die Objekte in einem S3-on-Outposts-Bucket auflisten.

Verwendung der AWS CLI

Im folgenden Beispiel werden die Objekte in einem S3-on-Outposts-Bucket (`s3-outposts:ListObjectsV2`) unter Verwendung der AWS CLI aufgelistet. Zum Verwenden dieses Befehls ersetzen Sie *`user input placeholder`* durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter list-objects-v2 in der AWS CLI-Referenz.

```
aws s3api list-objects-v2 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point
```

Note

Wenn Sie diese Aktion mit Amazon S3 auf Outposts über die AWS SDKs verwenden, geben Sie den Outposts-Zugangspunkt-ARN anstelle des Bucket-Namens in der folgenden Form an: `arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-Outposts-Access-Point`. Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Verwenden des AWS-SDKs für Java

Das folgende S3-on-Outposts-Beispiel listet Objekte in einem Bucket mit dem SDK for Java auf. Zum Verwenden dieses Beispiels ersetzen Sie jeden *`user input placeholder`* durch Ihre eigenen Informationen.

> **Important**
>
> In diesem Beispiel verwenden wir ListObjectsV2, welches die neueste Version der `ListObjects`-API-Operation ist. Wir empfehlen die Verwendung dieser überarbeiteten API-Operationen für die Anwendungsentwicklung. Aus Gründen der Abwärtskompatibilität unterstützt Amazon S3 weiterhin die vorherige Version dieser API-Operation.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsV2Request;
import com.amazonaws.services.s3.model.ListObjectsV2Result;
import com.amazonaws.services.s3.model.S3ObjectSummary;

public class ListObjectsV2 {

    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            System.out.println("Listing objects");

            // maxKeys is set to 2 to demonstrate the use of
            // ListObjectsV2Result.getNextContinuationToken()
            ListObjectsV2Request req = new
 ListObjectsV2Request().withBucketName(accessPointArn).withMaxKeys(2);
            ListObjectsV2Result result;

            do {
                result = s3Client.listObjectsV2(req);

                for (S3ObjectSummary objectSummary : result.getObjectSummaries()) {
```

```
                    System.out.printf(" - %s (size: %d)\n", objectSummary.getKey(),
 objectSummary.getSize());
                }
                // If there are more than maxKeys keys in the bucket, get a
 continuation token
                // and list the next objects.
                String token = result.getNextContinuationToken();
                System.out.println("Next Continuation Token: " + token);
                req.setContinuationToken(token);
            } while (result.isTruncated());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Löschen von Objekten in Amazon-S3-on.Outposts-Buckets

Objekte sind die Grundeinheiten, die in Amazon S3 on Outposts gespeichert sind. Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Amazon-Ressourcennamen (ARN) oder den Alias des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Objekte, das den AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, Outpost-ID, Bucket-Name und den Namen des Zugriffspunkts umfasst:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Die folgenden Beispiele veranschaulichen, wie Sie ein einzelnes Objekt oder mehrere Objekte in einem S3-on-Outposts-Bucket mithilfe der AWS Command Line Interface (AWS CLI) und AWS SDK for Java löschen.

Verwendung der AWS CLI

Die folgenden Beispiele veranschaulichen, wie Sie ein einzelnes Objekt oder mehrere Objekte aus einem S3-on-Outposts-Bucket löschen.

delete-object

Im folgenden Beispiel wird ein Objekt mit dem Namen `sample-object.xml` in einem S3-on-Outposts-Bucket (`s3-outposts:DeleteObject`) mithilfe der AWS CLI gelöscht. Zum Verwenden dieses Befehls ersetzen Sie *`user input placeholder`* durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter delete-object in der AWS CLI-Befehlsreferenz.

```
aws s3api delete-object --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml
```

delete-objects

Im folgenden Beispiel werden zwei Objekte mit dem Namen `sample-object.xml` und `test1.text` in einem S3-on-Outposts-Bucket (`s3-outposts:DeleteObject`) mithilfe der AWS CLI gelöscht. Zum Verwenden dieses Befehls ersetzen Sie *`user input placeholder`* durch Ihre eigenen Informationen. Weitere Informationen zu diesem Befehl finden Sie unter delete-objects in der AWS CLI-Referenz.

```
aws s3api delete-objects --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --delete file://delete.json

delete.json
{
  "Objects": [
    {
      "Key": "test1.txt"
    },
    {
      "Key": "sample-object.xml"
    }
  ],
  "Quiet": false
}
```

Verwenden des AWS-SDKs für Java

Die folgenden Beispiele veranschaulichen, wie Sie ein einzelnes Objekt oder mehrere Objekte aus einem S3-on-Outposts-Bucket löschen.

DeleteObject

Im folgenden Beispiel für S3 on Outposts wird ein Objekt in einem Bucket mit dem SDK for Java gelöscht. Zum Verwenden dieses Beispiels geben Sie den Zugriffspunkt-ARN für den Outpost und den Schlüsselnamen für das Objekt an, das Sie löschen möchten. Weitere Informationen finden Sie unter DeleteObject in der API-Referenz zum Amazon Simple Storage Service.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectRequest;

public class DeleteObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** key name ****";

        try {
```

```
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(accessPointArn, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

DeleteObjects

Das folgende S3-on-Outposts-Beispiel lädt Objekte in einem Bucket hoch und löscht sie dann mithilfe des SDK for Java. Wenn Sie dieses Beispiel verwenden möchten, geben Sie den Zugriffspunkt-ARN für den Outpost an. Weitere Informationen finden Sie unter DeleteObjects in der API-Referenz zu Amazon Simple Storage Service.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectsRequest;
import com.amazonaws.services.s3.model.DeleteObjectsRequest.KeyVersion;
import com.amazonaws.services.s3.model.DeleteObjectsResult;

import java.util.ArrayList;

public class DeleteObjects {

    public static void main(String[] args) {
       String accessPointArn = "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point";
```

```
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Upload three sample objects.
            ArrayList<KeyVersion> keys = new ArrayList<KeyVersion>();
            for (int i = 0; i < 3; i++) {
                String keyName = "delete object example " + i;
                s3Client.putObject(accessPointArn, keyName, "Object number " + i + "
 to be deleted.");
                keys.add(new KeyVersion(keyName));
            }
            System.out.println(keys.size() + " objects successfully created.");

            // Delete the sample objects.
            DeleteObjectsRequest multiObjectDeleteRequest = new
 DeleteObjectsRequest(accessPointArn)
                    .withKeys(keys)
                    .withQuiet(false);

            // Verify that the objects were deleted successfully.
            DeleteObjectsResult delObjRes =
 s3Client.deleteObjects(multiObjectDeleteRequest);
            int successfulDeletes = delObjRes.getDeletedObjects().size();
            System.out.println(successfulDeletes + " objects successfully
 deleted.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwenden von HeadBucket, um festzustellen, ob ein S3-on-Outposts-Bucket vorhanden ist und Sie Zugriffsberechtigungen haben

Objekte sind die Grundeinheiten, die in Amazon S3 on Outposts gespeichert sind. Jedes Objekt ist in einem Bucket enthalten. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie den Bucket für Objektoperationen angeben, verwenden Sie den Amazon-Ressourcennamen (ARN) oder den Alias des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Das folgende Beispiel zeigt das ARN-Format für S3-on-Outposts-Objekte, das den AWS-Region-Code für die Region, in der sich der Outpost befindet, die AWS-Konto-ID, Outpost-ID, Bucket-Name und den Namen des Zugriffspunkts umfasst:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Weitere Informationen zu S3-in-Outposts-ARNs finden Sie unter Ressourcen-ARNs für S3 on Outposts.

> **Note**
>
> Mit Amazon S3 on Outposts werden Objektdaten immer im Outpost gespeichert. Wenn AWS ein Outpost-Rack installiert, bleiben Ihre Daten in Ihrem Outpost lokal, um die Anforderungen an die Datenspeicherorte zu erfüllen. Ihre Objekte verlassen niemals Ihren Outpost und befinden sich nicht in einer AWS-Region. Da die AWS Management Console innerhalb der Region gehostet wird, können Sie die Konsole nicht verwenden, um Objekte in Ihrem Outpost hochzuladen oder zu verwalten. Sie können die REST-API, AWS Command Line Interface (AWS CLI) und AWS-SDKs zum Hochladen und Verwalten Ihrer Objekte über Ihre Zugriffspunkte verwenden.

Die folgenden Beispiele für AWS Command Line Interface (AWS CLI) und AWS SDK for Java veranschaulichen, wie Sie die API-Operation HeadBucket verwenden, um festzustellen, ob ein Amazon-S3-on-Outpost-Bucket vorhanden ist und ob Sie Zugriffsberechtigungen für diesen Bucket besitzen. Weitere Informationen finden Sie unter HeadBucket in der API-Referenz zu Amazon Simple Storage Service.

Verwendung der AWS CLI

Im folgenden AWS CLI-Beispiel für S3 on Outposts wird der Befehl `head-bucket` verwendet, um festzustellen, ob ein Bucket vorhanden ist und ob Sie Zugriffsberechtigungen für diesen Bucket besitzen. Zum Verwenden dieses Befehls ersetzen Sie *`user input placeholder`* durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter head-bucket in der AWS CLI-Referenz.

```
aws s3api head-bucket --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

Verwenden des AWS-SDKs für Java

Das folgende S3-on-Outposts-Beispiel veranschaulicht, wie Sie feststellen, ob ein Bucket vorhanden ist und ob Sie Zugriffsberechtigungen für diesen Bucket besitzen. Wenn Sie dieses Beispiel verwenden möchten, geben Sie den Zugriffspunkt-ARN für den Outpost an. Weitere Informationen finden Sie unter HeadBucket in der API-Referenz zu Amazon Simple Storage Service.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.HeadBucketRequest;

public class HeadBucket {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.headBucket(new HeadBucketRequest(accessPointArn));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
```

```
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Durchführen und Verwalten eines mehrteiligen Uploads mit dem SDK for Java

Mit Amazon S3 in Outposts können Sie S3-Buckets in Ihren AWS Outposts-Ressourcen erstellen und Objekte On-Premises für Anwendungen speichern und abrufen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Die folgenden Beispiele veranschaulichen, wie Sie S3 on Outposts mit der AWS SDK for Java verwenden können, um einen mehrteiligen Upload durchzuführen und zu verwalten.

Themen

- Durchführen eines mehrteiligen Uploads eines Objekts in einem S3-on-Outposts-Bucket
- Kopieren eines großen Objekts in einem S3-on-Outposts-Bucket mithilfe eines mehrteiligen Uploads
- Auflisten von Teilen eines Objekts in einem S3-on-Outposts-Bucket
- Abrufen einer Liste der in Bearbeitung befindlichen mehrteiligen Uploads in einem S3-on-Outposts-Bucket

Durchführen eines mehrteiligen Uploads eines Objekts in einem S3-on-Outposts-Bucket

Das folgende S3-on-Outposts-Beispiel initiiert, lädt und beendet einen mehrteiligen Upload eines Objekts in einen Bucket mithilfe des SDK for Java. Zum Verwenden dieses Beispiels ersetzen Sie jeden *`user input placeholder`* durch Ihre eigenen Informationen. Weitere Informationen finden Sie unter Hochladen eines Objekts mit Multipart-Upload.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
```

```
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
            ObjectMetadata metadataResult =
 s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);
```

```
                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
```

Kopieren eines großen Objekts in einem S3-on-Outposts-Bucket mithilfe eines mehrteiligen Uploads

Im folgenden Beispiel wird ein Objekt mithilfe des SDK for Java in einem S3-on-Outposts-Bucket kopiert. Zum Verwenden dieses Beispiels ersetzen Sie jeden *user input placeholder* durch Ihre eigenen Informationen. Dieses Beispiel basiert auf Kopieren eines Objekts mit Multipart-Upload.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
            ObjectMetadata metadataResult =
 s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();
```

```
            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
}
```

Auflisten von Teilen eines Objekts in einem S3-on-Outposts-Bucket

Das folgende S3-on-Outposts-Beispiel listet die Teile eines Objekts in einem Bucket mit dem SDK for Java auf. Zum Verwenden dieses Beispiels ersetzen Sie jeden *user input placeholder* durch Ihre eigenen Informationen.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.List;

public class ListParts {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** Key name ***";
        String uploadId = "*** Upload ID ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            ListPartsRequest listPartsRequest = new ListPartsRequest(accessPointArn,
 keyName, uploadId);
            PartListing partListing = s3Client.listParts(listPartsRequest);
            List<PartSummary> partSummaries = partListing.getParts();
```

```
            System.out.println(partSummaries.size() + " multipart upload parts");
            for (PartSummary p : partSummaries) {
                System.out.println("Upload part: Part number = \"" + p.getPartNumber()
 + "\", ETag = " + p.getETag());
            }

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Abrufen einer Liste der in Bearbeitung befindlichen mehrteiligen Uploads in einem S3-on-Outposts-Bucket

Das folgende S3-on-Outposts-Beispiel zeigt, wie Sie mit dem SDK for Java eine Liste der laufenden mehrteiligen Uploads aus einem Outposts-Bucket abrufen. Zum Verwenden dieses Beispiels ersetzen Sie jeden *`user input placeholder`* durch Ihre eigenen Informationen. Dies ist ein Beispiel, das aus dem Beispiel für Auflisten von mehrteiligen Uploads für Amazon S3 adaptiert wurde.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListMultipartUploadsRequest;
import com.amazonaws.services.s3.model.MultipartUpload;
import com.amazonaws.services.s3.model.MultipartUploadListing;

import java.util.List;

public class ListMultipartUploads {
    public static void main(String[] args) {
                String accessPointArn = "*** access point ARN ***";

        try {
```

```
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Retrieve a list of all in-progress multipart uploads.
            ListMultipartUploadsRequest allMultipartUploadsRequest = new
 ListMultipartUploadsRequest(accessPointArn);
            MultipartUploadListing multipartUploadListing =
 s3Client.listMultipartUploads(allMultipartUploadsRequest);
            List<MultipartUpload> uploads =
 multipartUploadListing.getMultipartUploads();

            // Display information about all in-progress multipart uploads.
            System.out.println(uploads.size() + " multipart upload(s) in progress.");
            for (MultipartUpload u : uploads) {
                System.out.println("Upload in progress: Key = \"" + u.getKey() + "\",
 id = " + u.getUploadId());
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwenden vorsignierter URLs für S3 on Outposts

Um zeitlich begrenzten Zugriff auf Objekte zu gewähren, die lokal auf einem Outpost gespeichert sind, ohne Ihre Bucket-Richtlinie zu aktualisieren, können Sie eine vordefinierte URL verwenden. Mit vorsignierten URLs können Sie als Bucket-Besitzer Objekte für Personen in Ihrer Virtual Private Cloud (VPC) freigeben oder ihnen die Möglichkeit geben, Objekte hochzuladen oder zu löschen.

Wenn Sie eine vorsignierte URL mit Hilfe der AWS-SDKs oder AWS Command Line Interface (AWS CLI) erstellen, verknüpfen Sie die URL mit einer bestimmten Aktion. Sie können auch einen

zeitlich begrenzten Zugriff auf die vorsignierte URL gewähren, indem Sie eine benutzerdefinierte Ablaufzeit wählen, die zwischen 1 Sekunde und 7 Tagen liegen kann. Wenn Sie die vorsignierte URL freigeben, kann die Person in der VPC die in der URL eingebettete Aktion so ausführen, als wäre sie der ursprünglich signierende Benutzer. Wenn die URL ihre Verfallszeit erreicht, läuft sie ab und funktioniert nicht mehr.

Beschränkung der Funktionen für vorsignierte URLs

Die Funktionen einer vorsignierten URL sind durch die Berechtigungen des Benutzers eingeschränkt, der sie erstellt hat. Im Wesentlichen sind vorsignierte URLs Inhaber-Token, die denjenigen, die sie besitzen, Zugriff gewähren. Daher empfehlen wir Ihnen, sie angemessen zu schützen.

AWS Signature Version 4 (SigV4)

Um ein bestimmtes Verhalten zu erzwingen, wenn vorsignierte URL-Anfragen mit AWS Signature Version 4 (SigV4) authentifiziert werden, können Sie Bedingungsschlüssel in Bucket-Richtlinien und Zugriffspunkt-Richtlinien verwenden. Sie können z. B. eine Bucket-Richtlinie erstellen, die die `s3-outposts:signatureAge`-Bedingung verwendet, um jede vorsignierte URL-Anfrage von Amazon S3 on Outposts für Objekte im `example-outpost-bucket`-Bucket zu verweigern, wenn die Signatur mehr als 10 Minuten alt ist. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

Eine Liste von Bedingungsschlüsseln und zusätzlichen Beispielrichtlinien, die Sie verwenden können, um ein bestimmtes Verhalten zu erzwingen, wenn vorsignierte URL-Anfragen mit Hilfe von Signature Version 4 authentifiziert werden, finden Sie unter AWS-Signature Version 4 (SigV4) – Authentifizierungsspezifische Richtlinienschlüssel.

Beschränkung der Netzwege

Wenn Sie die Verwendung von vorsignierten URLs und den Zugriff aller S3 on Outposts auf bestimmte Netzwerkpfade beschränken möchten, können Sie Richtlinien schreiben, die einen bestimmten Netzwerkpfad erfordern. Um die Beschränkung auf den IAM-Prinzipal festzulegen, der den Anruf tätigt, können Sie identitätsbasierte AWS Identity and Access Management (IAM)-Richtlinien verwenden (z. B. Benutzer-, Gruppen- oder Rollenrichtlinien). Um die Beschränkung für die Ressource S3 on Outposts festzulegen, können Sie ressourcenbasierte Richtlinien verwenden (z. B. Bucket- und Zugriffspunkt-Richtlinien).

Eine Netzwerkpfadbeschränkung für den IAM-Prinzipal erfordert, dass der Benutzer dieser Anmeldeinformationen Anfragen aus dem angegebenen Netzwerk stellt. Eine Einschränkung des Buckets oder des Zugriffspunkts erfordert, dass alle Anfragen an diese Ressource aus dem angegebenen Netz stammen. Diese Einschränkungen gelten auch außerhalb des Szenarios der vorsignierten URL.

Die globale IAM-Bedingung, die Sie verwenden, hängt von der Art des Endpunkts ab. Wenn Sie den öffentlichen Endpunkt für S3 on Outposts verwenden, benutzen Sie `aws:SourceIp`. Wenn Sie einen VPC-Endpunkt für S3 on Outposts verwenden, verwenden Sie `aws:SourceVpc` oder `aws:SourceVpce`.

Die folgende IAM-Richtlinienanweisung verlangt, dass der Prinzipal ausschließlich über den angegebenen Netzwerkbereich auf AWS zugreift. Mit dieser Richtlinie müssen alle Zugriffe von diesem Bereich ausgehen. Dies gilt auch für den Fall, dass jemand eine vorsignierte URL für S3 on Outposts verwendet. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

```
{
    "Sid": "NetworkRestrictionForIAMPrincipal",
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
        "NotIpAddressIfExists": {"aws:SourceIp": "IP-address-range"},
        "BoolIfExists": {"aws:ViaAWSService": "false"}
```

```
    }
}
```

Ein Beispiel für eine Bucket-Richtlinie, die den `aws:SourceIP` AWS globalen Bedingungsschlüssel verwendet, um den Zugriff auf einen S3 on Outposts-Bucket auf einen bestimmten Netzwerkbereich zu beschränken, finden Sie unter Einrichten von IAM mit S3 on Outposts.

Wer eine vorsignierte URL erstellen kann

Alle Benutzer mit gültigen Sicherheitsanmeldeinformationen können vorsignierte URLs erstellen. Damit ein Benutzer in der VPC jedoch erfolgreich auf ein Objekt zugreifen kann, muss die zugewiesene URL von jemandem erstellt werden, der die Berechtigung hat, den Vorgang durchzuführen, auf dem die zugewiesene URL basiert.

Sie können die folgenden Anmeldeinformationen verwenden, um eine vorsignierte URL zu erstellen:

- IAM-Instance-Profil – Bis zu 6 Stunden gültig.
- AWS Security Token Service – Gültig bis zu 36 Stunden, wenn mit dauerhaften Anmeldeinformationen signiert wird, z. B. mit den Anmeldeinformationen des AWS-Konto Stammbenutzers oder eines IAM-Benutzers.
- IAM-Benutzer - Gültig bis zu 7 Tage, wenn Sie die AWS Signature Version 4 verwenden.

  Um eine vordefinierte URL zu erstellen, die bis zu 7 Tage gültig ist, delegieren Sie zunächst die IAM-Benutzer-Anmeldeinformationen (den Zugriffsschlüssel und den geheimen Schlüssel) an das von Ihnen verwendete SDK. Erzeugen Sie dann eine vorsignierte URL, indem Sie AWS Signature Version 4 verwenden.

> **Note**
>
> - Wenn Sie eine vorsignierte URL mit einem temporären Token erstellt haben, läuft die URL ab, wenn das Token abläuft, auch wenn Sie die URL mit einer späteren Ablaufzeit erstellt haben.
> - Da vorsignierte URLs jedem, der über die URL verfügt, Zugriff auf Ihre S3 on Outposts-Buckets gewähren, empfehlen wir Ihnen, diese entsprechend zu schützen. Weitere Informationen zum Schutz von vorsignierten URLs finden Sie unter Beschränkung der Funktionen für vorsignierte URLs.

Wann prüft S3 on Outposts das Ablaufdatum und die Uhrzeit einer vorsignierten URL?

Zum Zeitpunkt der HTTP-Anfrage überprüft S3 on Outposts das Ablaufdatum und die Uhrzeit einer signierten URL. Beginnt ein Client beispielsweise mit dem Herunterladen einer großen Datei unmittelbar vor der Ablaufzeit, wird der Download auch dann fortgesetzt, wenn die Ablaufzeit während des Downloads verstreicht. Wenn die Verbindung jedoch unterbrochen wird und der Client versucht, den Download nach Ablauf der Zeit erneut zu starten, schlägt der Download fehl.

Weitere Informationen zur Verwendung einer vorsignierten URL zum Teilen oder Hochladen von Objekten finden Sie in den folgenden Themen.

Themen

- Gemeinsame Nutzung von Objekten unter Verwendung vorsignierter URLs
- Generierung einer vorsignierten URL zum Hochladen eines Objekts in einen S3 on Outposts-Bucket

Gemeinsame Nutzung von Objekten unter Verwendung vorsignierter URLs

Um zeitlich begrenzten Zugriff auf Objekte zu gewähren, die lokal auf einem Outpost gespeichert sind, ohne Ihre Bucket-Richtlinie zu aktualisieren, können Sie eine vorsignierte URL verwenden. Mit vorsignierten URLs können Sie als Bucket-Besitzer Objekte für Personen in Ihrer Virtual Private Cloud (VPC) freigeben oder ihnen die Möglichkeit geben, Objekte hochzuladen oder zu löschen.

Wenn Sie eine vorsignierte URL mit Hilfe der AWS-SDKs oder AWS Command Line Interface (AWS CLI) erstellen, verknüpfen Sie die URL mit einer bestimmten Aktion. Sie können auch einen zeitlich begrenzten Zugriff auf die vorsignierte URL gewähren, indem Sie eine benutzerdefinierte Ablaufzeit wählen, die zwischen 1 Sekunde und 7 Tagen liegen kann. Wenn Sie die vorsignierte URL freigeben, kann die Person in der VPC die in der URL eingebettete Aktion so ausführen, als wäre sie der ursprünglich signierende Benutzer. Wenn die URL ihre Verfallszeit erreicht, läuft sie ab und funktioniert nicht mehr.

Wenn Sie eine vorsignierte URL erstellen, müssen Sie Ihre Sicherheitsanmeldedaten eingeben und dann Folgendes angeben:

- Ein Zugriffspunkt Amazon-Ressourcenname (ARN) für den Amazon S3 on Outposts Bucket
- Ein Objektschlüssel
- Eine HTTP-Methode (`GET` zum Herunterladen von Objekten)

- Ein Verfallsdatum und eine Verfallszeit

Eine vorsignierte URL ist nur für die angegebene Dauer gültig. Das heißt, Sie müssen die von der URL erlaubte Aktion vor dem Ablaufdatum und der Ablaufzeit starten. Sie können eine vorsignierte URL bis zum Ablaufdatum und zur Ablaufzeit mehrfach verwenden. Wenn Sie eine vorsignierte URL mit einem temporären Token erstellt haben, läuft die URL ab, wenn das Token abläuft, auch wenn Sie die URL mit einer späteren Ablaufzeit erstellt haben.

Benutzer in der Virtual Private Cloud (VPC), die Zugriff auf die vorsignierte URL haben, können auf das Objekt zugreifen. Wenn Sie beispielsweise ein Video in Ihrem Bucket haben und sowohl der Bucket als auch das Objekt privat sind, können Sie das Video mit anderen teilen, indem Sie eine vorsignierte URL generieren. Da vorsignierte URLs jedem, der über die URL verfügt, Zugriff auf Ihre S3 on Outposts-Buckets gewähren, empfehlen wir Ihnen, diese URLs entsprechend zu schützen. Weitere Informationen zum Schutz vorsignierter URLs finden Sie unter Beschränkung der Funktionen für vorsignierte URLs.

Alle Benutzer mit gültigen Sicherheitsanmeldeinformationen können vorsignierte URLs erstellen. Die vorsignierte URL muss jedoch von jemandem erstellt werden, der die Berechtigung hat, den Vorgang durchzuführen, auf dem die vorsignierte URL basiert. Weitere Informationen finden Sie unter Wer eine vorsignierte URL erstellen kann.

Sie können eine vorsignierte URL zur Freigabe eines Objekts in einem S3 on Outposts-Bucket generieren, indem Sie die AWS-SDKs und die AWS CLI anwenden. Weitere Informationen finden Sie in den folgenden Beispielen.

Verwenden der AWS-SDKs

Sie können die AWS-SDKs verwenden, um eine vorsignierte URL zu generieren, die Sie an andere weitergeben können, damit diese ein Objekt abrufen können.

> **Note**
>
> Wenn Sie die AWS-SDKs verwenden, um eine vorsignierte URL zu erzeugen, beträgt die maximale Verfallszeit für eine vorsignierte URL 7 Tage ab dem Zeitpunkt der Erstellung.

Java

Example

Das folgende Beispiel generiert eine vorsignierte URL, die Sie an andere weitergeben können, damit diese ein Objekt aus einem S3 on Outposts-Bucket abrufen können. Weitere Informationen finden Sie unter Verwenden vorsignierter URLs für S3 on Outposts. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Java-Codebeispiele für Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.HttpMethod;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GeneratePresignedUrlRequest;

import java.io.IOException;
import java.net.URL;
import java.time.Instant;

public class GeneratePresignedURL {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String accessPointArn = "*** access point ARN ***";
        String objectKey = "*** object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Set the presigned URL to expire after one hour.
            java.util.Date expiration = new java.util.Date();
            long expTimeMillis = Instant.now().toEpochMilli();
            expTimeMillis += 1000 * 60 * 60;
            expiration.setTime(expTimeMillis);
```

```
            // Generate the presigned URL.
            System.out.println("Generating pre-signed URL.");
            GeneratePresignedUrlRequest generatePresignedUrlRequest =
                    new GeneratePresignedUrlRequest(accessPointArn, objectKey)
                            .withMethod(HttpMethod.GET)
                            .withExpiration(expiration);
            URL url = s3Client.generatePresignedUrl(generatePresignedUrlRequest);

            System.out.println("Pre-Signed URL: " + url.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't
 process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Example

Das folgende Beispiel generiert eine vorsignierte URL, die Sie an andere weitergeben können, damit diese ein Objekt aus einem S3 on Outposts-Bucket abrufen können. Weitere Informationen finden Sie unter Verwenden vorsignierter URLs für S3 on Outposts. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der .NET-Codebeispiele für Amazon S3.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;

namespace Amazon.DocSamples.S3
{
    class GenPresignedURLTest
```

```
    {
        private const string accessPointArn = "*** access point ARN ***";
        private const string objectKey = "*** object key ***";
        // Specify how long the presigned URL lasts, in hours.
        private const double timeoutDuration = 12;
        // Specify your bucket Region (an example Region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            string urlString = GeneratePreSignedURL(timeoutDuration);
        }
        static string GeneratePreSignedURL(double duration)
        {
            string urlString = "";
            try
            {
                GetPreSignedUrlRequest request1 = new GetPreSignedUrlRequest
                {
                    BucketName = accessPointArn,
                    Key = objectKey,
                    Expires = DateTime.UtcNow.AddHours(duration)
                };
                urlString = s3Client.GetPreSignedURL(request1);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            return urlString;
        }
    }
}
```

Python

Das folgende Beispiel generiert eine vorsignierte URL zur Freigabe eines Objekts mit Hilfe des SDK für Python (Boto3). Verwenden Sie z. B. einen Boto3-Client und die `generate_presigned_url` Funktion, um eine vorsignierte URL zu generieren, die Ihnen ermöglicht zu `GET` ein Objekt.

```
import boto3
    url = boto3.client('s3').generate_presigned_url(
    ClientMethod='get_object',
    Params={'Bucket': 'ACCESS_POINT_ARN', 'Key': 'OBJECT_KEY'},
    ExpiresIn=3600)
```

Weitere Informationen zur Verwendung von SDK for Python (Boto3) zur Erzeugung einer vorsignierten URL finden Sie unter Python in der API–Referenz für AWS SDK for Python (Boto).

Verwendung der AWS CLI

Der folgende AWS CLI Beispielbefehl generiert eine vorsignierte URL für einen S3 on Outposts-Bucket. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

> **Note**
>
> Wenn Sie die Funktion AWS CLI verwenden, um eine vorsignierte URL zu erstellen, beträgt die maximale Verfallszeit für eine vorsignierte URL 7 Tage ab dem Zeitpunkt der Erstellung.

```
aws s3 presign s3://arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-
point/mydoc.txt --expires-in 604800
```

Weitere Informationen finden Sie unter vorsignieren in der AWS CLIBefehlsreferenz.

Generierung einer vorsignierten URL zum Hochladen eines Objekts in einen S3 on Outposts-Bucket

Um zeitlich begrenzten Zugriff auf Objekte zu gewähren, die lokal auf einem Outpost gespeichert sind, ohne Ihre Bucket-Richtlinie zu aktualisieren, können Sie eine vorsignierte URL verwenden.

Mit vorsignierten URLs können Sie als Bucket-Besitzer Objekte für Personen in Ihrer Virtual Private Cloud (VPC) freigeben oder ihnen die Möglichkeit geben, Objekte hochzuladen oder zu löschen.

Wenn Sie eine vorsignierte URL mit Hilfe der AWS-SDKs oder AWS Command Line Interface (AWS CLI) erstellen, verknüpfen Sie die URL mit einer bestimmten Aktion. Sie können auch einen zeitlich begrenzten Zugriff auf die vorsignierte URL gewähren, indem Sie eine benutzerdefinierte Ablaufzeit wählen, die zwischen 1 Sekunde und 7 Tagen liegen kann. Wenn Sie die vorsignierte URL freigeben, kann die Person in der VPC die in der URL eingebettete Aktion so ausführen, als wäre sie der ursprünglich signierende Benutzer. Wenn die URL ihre Verfallszeit erreicht, läuft sie ab und funktioniert nicht mehr.

Wenn Sie eine vorsignierte URL erstellen, müssen Sie Ihre Sicherheitsanmeldedaten eingeben und dann Folgendes angeben:

- Ein Zugriffspunkt Amazon-Ressourcenname (ARN) für den Amazon S3 on Outposts Bucket
- Ein Objektschlüssel
- Eine HTTP-Methode (PUT zum Hochladen von Objekten)
- Ein Verfallsdatum und eine Verfallszeit

Eine vorsignierte URL ist nur für die angegebene Dauer gültig. Das heißt, Sie müssen die von der URL erlaubte Aktion vor dem Ablaufdatum und der Ablaufzeit starten. Sie können eine vorsignierte URL bis zum Ablaufdatum und zur Ablaufzeit mehrfach verwenden. Wenn Sie eine vorsignierte URL mit einem temporären Token erstellt haben, läuft die URL ab, wenn das Token abläuft, auch wenn Sie die URL mit einer späteren Ablaufzeit erstellt haben.

Wenn die von einer vorsignierten URL erlaubte Aktion aus mehreren Schritten besteht, wie z. B. ein mehrteiliger Upload, müssen Sie alle Schritte vor Ablauf der Zeit starten. Wenn S3 on Outposts versucht, einen Schritt mit einer abgelaufenen URL zu starten, erhalten Sie eine Fehlermeldung.

Benutzer in der Virtual Private Cloud (VPC), die Zugriff auf die vorsignierte URL haben, können Objekte hochladen. So kann beispielsweise ein Benutzer in der VPC, der Zugriff auf die vorsignierte URL hat, ein Objekt in Ihren Bucket hochladen. Da vorsignierte URLs jedem Benutzer in der VPC, der Zugriff auf die vorsignierte URL hat, Zugriff auf Ihren S3 on Outposts-Bucket gewähren, empfehlen wir Ihnen, diese URLs entsprechend zu schützen. Weitere Informationen zum Schutz vorsignierter URLs finden Sie unter Beschränkung der Funktionen für vorsignierte URLs.

Alle Benutzer mit gültigen Sicherheitsanmeldeinformationen können vorsignierte URLs erstellen. Die vorsignierte URL muss jedoch von jemandem erstellt werden, der die Berechtigung hat, den Vorgang

durchzuführen, auf dem die vorsignierte URL basiert. Weitere Informationen finden Sie unter Wer eine vorsignierte URL erstellen kann.

Verwendung der AWS-SDKs zur Generierung einer vorsignierten URL für eine S3 on Outposts-Objektoperation

Java

SDK für Java 2.x

Dieses Beispiel zeigt, wie Sie eine vorsignierte URL generieren, mit der Sie ein Objekt für eine begrenzte Zeit in einen S3 on Outposts-Bucket hochladen können. Weitere Informationen finden Sie unter Verwenden vorsignierter URLs für S3 on Outposts.

```
   public static void signBucket(S3Presigner presigner, String
outpostAccessPointArn, String keyName) {

       try {
           PutObjectRequest objectRequest = PutObjectRequest.builder()
                   .bucket(accessPointArn)
                   .key(keyName)
                   .contentType("text/plain")
                   .build();

           PutObjectPresignRequest presignRequest =
PutObjectPresignRequest.builder()
                   .signatureDuration(Duration.ofMinutes(10))
                   .putObjectRequest(objectRequest)
                   .build();

           PresignedPutObjectRequest presignedRequest =
presigner.presignPutObject(presignRequest);


           String myURL = presignedRequest.url().toString();
           System.out.println("Presigned URL to upload a file to: " +myURL);
           System.out.println("Which HTTP method must be used when uploading a
file: " +
                   presignedRequest.httpRequest().method());

           // Upload content to the S3 on Outposts bucket by using this URL.
           URL url = presignedRequest.url();
```

```
            // Create the connection and use it to upload the new object by using
the presigned URL.
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
            connection.setDoOutput(true);
            connection.setRequestProperty("Content-Type","text/plain");
            connection.setRequestMethod("PUT");
            OutputStreamWriter out = new
OutputStreamWriter(connection.getOutputStream());
            out.write("This text was uploaded as an object by using a presigned
URL.");
            out.close();

            connection.getResponseCode();
            System.out.println("HTTP response code is " +
connection.getResponseCode());

        } catch (S3Exception e) {
            e.getStackTrace();
        } catch (IOException e) {
            e.getStackTrace();
        }
    }
```

Python

SDK für Python (Boto3)

In diesem Beispiel wird gezeigt, wie man eine vorsignierte URL generiert, die für eine begrenzte Zeit eine S3 on Outposts-Aktion ausführen kann. Weitere Informationen finden Sie unter Verwenden vorsignierter URLs für S3 on Outposts. Um eine Anfrage mit der URL zu stellen, verwenden Sie das `Requests` Paket.

```
import argparse
import logging
import boto3
from botocore.exceptions import ClientError
import requests

logger = logging.getLogger(__name__)
```

```
def generate_presigned_url(s3_client, client_method, method_parameters,
 expires_in):
    """
    Generate a presigned S3 on Outposts URL that can be used to perform an
 action.

    :param s3_client: A Boto3 Amazon S3 client.
    :param client_method: The name of the client method that the URL performs.
    :param method_parameters: The parameters of the specified client method.
    :param expires_in: The number of seconds that the presigned URL is valid for.
    :return: The presigned URL.
    """
    try:
        url = s3_client.generate_presigned_url(
            ClientMethod=client_method,
            Params=method_parameters,
            ExpiresIn=expires_in
        )
        logger.info("Got presigned URL: %s", url)
    except ClientError:
        logger.exception(
            "Couldn't get a presigned URL for client method '%s'.",
 client_method)
        raise
    return url


def usage_demo():
    logging.basicConfig(level=logging.INFO, format='%(levelname)s: %(message)s')

    print('-'*88)
    print("Welcome to the Amazon S3 on Outposts presigned URL demo.")
    print('-'*88)

    parser = argparse.ArgumentParser()
    parser.add_argument('accessPointArn', help="The name of the S3 on Outposts
 access point ARN.")
    parser.add_argument(
        'key', help="For a GET operation, the key of the object in S3 on
 Outposts. For a "
                    "PUT operation, the name of a file to upload.")
    parser.add_argument(
        'action', choices=('get', 'put'), help="The action to perform.")
    args = parser.parse_args()
```

```
    s3_client = boto3.client('s3')
    client_action = 'get_object' if args.action == 'get' else 'put_object'
    url = generate_presigned_url(
        s3_client, client_action, {'Bucket': args.accessPointArn, 'Key':
 args.key}, 1000)

    print("Using the Requests package to send a request to the URL.")
    response = None
    if args.action == 'get':
        response = requests.get(url)
    elif args.action == 'put':
        print("Putting data to the URL.")
        try:
            with open(args.key, 'r') as object_file:
                object_text = object_file.read()
            response = requests.put(url, data=object_text)
        except FileNotFoundError:
            print(f"Couldn't find {args.key}. For a PUT operation, the key must
 be the "
                  f"name of a file that exists on your computer.")

    if response is not None:
        print("Got response:")
        print(f"Status: {response.status_code}")
        print(response.text)

    print('-'*88)


if __name__ == '__main__':
    usage_demo()
```

Amazon S3 on Outposts mit lokalem Amazon EMR on Outposts

Amazon EMR ist eine verwaltete Cluster-Plattform, die die Ausführung von Big-Data-Frameworks wie Apache Hadoop und vereinfachtApache Spark, AWS um riesige Datenmengen zu verarbeiten und zu analysieren. Durch die Verwendung dieser Frameworks und verwandter Open-Source-Projekte können Sie Daten für Analysezwecke und Business-Intelligence-Workloads verarbeiten. Amazon EMR unterstützt Sie auch bei der Transformation und Übertragung großer Datenmengen in und aus anderen AWS Datenspeichern und Datenbanken und unterstützt Amazon S3 on Outposts. Weitere

Informationen zu Amazon EMR finden Sie unter Amazon EMR in Outposts im Verwaltungshandbuch für Amazon EMR.

Für Amazon S3 on Outposts begann Amazon EMR mit der Unterstützung des Apache Hadoop S3A-Konnektors in Version 7.0.0. Frühere Versionen von Amazon EMR unterstützen lokale S3 on Outposts nicht, und das EMR File System (EMRFS) wird nicht unterstützt.

Unterstützte Anwendungen

Amazon EMR mit Amazon S3 on Outposts unterstützt die folgenden Anwendungen:

- Hadoop
- Spark
- Hue
- Hive
- Sqoop
- Pig
- Hudi
- Flink

Weitere Informationen finden Sie im Handbuch zu Amazon-EMR-Versionen.

Erstellen und Konfigurieren eines Amazon S3-on-Outposts-Buckets

Amazon EMR verwendet die AWS SDK for Java mit Amazon S3 on Outposts, um Eingabe- und Ausgabedaten zu speichern. Ihre Amazon-EMR-Protokolldateien werden an einem von Ihnen ausgewählten regionalen Amazon S3-Speicherort gespeichert und nicht lokal auf dem Outpost gespeichert. Weitere Informationen finden Sie unter Amazon-EMR-Protokolle im Verwaltungshandbuch für Amazon EMR.

Um den Amazon S3- und DNS-Anforderungen zu entsprechen, haben S3-on-Outposts-Buckets Namensbeschränkungen und Einschränkungen. Weitere Informationen finden Sie unter Erstellen eines S3-on-Outposts-Buckets.

Mit Amazon-EMR-Version 7.0.0 und höher können Sie Amazon EMR mit S3 on Outposts und dem S3A-Dateisystem verwenden.

Voraussetzungen

Berechtigungen für S3 on Outposts – Wenn Sie Ihr Amazon-EMR-Instance-Profil erstellen, muss Ihre Rolle den AWS Identity and Access Management (IAM)-Namespace für S3 on Outposts enthalten. S3 on Outposts hat seinen eigenen Namespace, `s3-outposts*`. Eine Beispielrichtlinie, die diesen Namespace verwendet, finden Sie unter Einrichten von IAM mit S3 on Outposts.

S3A-Konnektor – Um Ihren EMR-Cluster für den Zugriff auf Daten aus einem Amazon S3-on-Outposts-Bucket zu konfigurieren, müssen Sie den Apache Hadoop S3A-Konnektor verwenden. Um den Konnektor zu verwenden, stellen Sie sicher, dass alle Ihre S3-URIs das `s3a` Schema verwenden. Andernfalls können Sie die Dateisystemimplementierung konfigurieren, die Sie für Ihren EMR-Cluster verwenden, sodass Ihre S3-URIs mit dem S3A-Konnektor funktionieren.

Um die Dateisystemimplementierung für die Arbeit mit dem S3A-Konnektor zu konfigurieren, verwenden Sie die `fs.AbstractFileSystem.`*`file_scheme`*`.impl` Konfigurationseigenschaften `fs.`*`file_scheme`*`.impl` und für Ihren EMR-Cluster, wobei dem Typ der S3-URIs *`file_scheme`* entspricht, die Sie haben. Wenn Sie das folgende Beispiel verwenden möchten, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen. Um beispielsweise die Dateisystemimplementierung für S3-URIs zu ändern, die das -s3Schema verwenden, geben Sie die folgenden Cluster-Konfigurationseigenschaften an:

```
[
  {
"Classification": "core-site",
    "Properties": {
    "fs.s3.impl": "org.apache.hadoop.fs.s3a.S3AFileSystem",
    "fs.AbstractFileSystem.s3.impl": "org.apache.hadoop.fs.s3a.S3A"
    }
  }
]
```

Um S3A zu verwenden, setzen Sie die `fs.`*`file_scheme`*`.impl` Konfigurationseigenschaft auf `org.apache.hadoop.fs.s3a.S3AFileSystem`und die `fs.AbstractFileSystem.`*`file_scheme`*`.impl` Eigenschaft auf `org.apache.hadoop.fs.s3a.S3A`.

Wenn Sie beispielsweise auf den Pfad zugreifen`s3a://`*`bucket`*`/...`, setzen Sie die `-fs.s3a.impl`Eigenschaft auf `org.apache.hadoop.fs.s3a.S3AFileSystem`und die `-fs.AbstractFileSystem.s3a.impl`Eigenschaft auf `org.apache.hadoop.fs.s3a.S3A`.

Erste Schritte mit Amazon EMR mit Amazon S3 on Outposts

In den folgenden Themen werden die ersten Schritte bei der Verwendung von Amazon EMR mit Amazon S3 on Outposts erläutert.

Themen

Erstellen einer Berechtigungsrichtlinie

Bevor Sie einen EMR-Cluster erstellen können, der Amazon S3 on Outposts verwendet, müssen Sie eine IAM-Richtlinie erstellen, die an das Amazon EC2-Instance-Profil für den Cluster angehängt wird. Die Richtlinie muss über Berechtigungen für den Zugriff auf den Amazon-Ressourcennamen (ARN) des S3-on-Outposts-Zugriffspunkts verfügen. Weitere Informationen zum Erstellen von IAM-Richtlinien für S3 on Outposts finden Sie unter Einrichten von IAM mit S3 on Outposts.

Die folgende Beispielrichtlinie zeigt, wie Sie die erforderlichen Berechtigungen erteilen. Nachdem Sie die Richtlinie erstellt haben, ordnen Sie die Richtlinie der Instance-Profilrolle zu, mit der Sie Ihren EMR-Cluster erstellen, wie im Abschnitt the section called “Ihren Cluster erstellen und konfigurieren” beschrieben. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
{
"Version":"2012-10-17",
  "Statement": [
        {
  "Effect": "Allow",
            "Resource": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name,
            "Action": [
                "s3-outposts:*"
            ]
        }
    ]

 }
```

Ihren Cluster erstellen und konfigurieren

Führen Sie die folgenden Schritte in der Konsole aus, um einen Cluster zu erstellen, auf dem Spark mit S3 on Outposts ausgeführt wird.

So erstellen Sie einen Cluster, der Spark mit S3 on Outposts ausgeführt wird

1. Öffnen Sie die Amazon-EMR-Konsole unterhttps://console.aws.amazon.com/elasticmapreduce/.
2. Wählen Sie im linken Navigationsbereich die Option Cluster aus.
3. Wählen Sie Cluster erstellen.
4. Wählen Sie für Amazon-EMR-Version emr-7.0.0 oder höher aus.
5. Wählen Sie für Anwendungspaket die Option Interaktive Spark- aus. Wählen Sie dann alle anderen unterstützten Anwendungen aus, die in Ihren Cluster aufgenommen werden sollen.
6. Um Amazon S3 on Outposts zu aktivieren, geben Sie Ihre Konfigurationseinstellungen ein.

   Beispielkonfigurationseinstellungen

   Um die folgenden Beispielkonfigurationseinstellungen zu verwenden, ersetzen Sie durch *user input placeholders* Ihre eigenen Informationen.

```
[
 {
   "Classification": "core-site",
   "Properties": {
     "fs.s3a.bucket.DOC-EXAMPLE-BUCKET.accesspoint.arn": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name"
     "fs.s3a.committer.name": "magic",
     "fs.s3a.select.enabled": "false"
    }
  },
  {
    "Classification": "hadoop-env",
    "Configurations": [
      {
        "Classification": "export",
        "Properties": {
          "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
          }
       }
     ],
```

```
      "Properties": {}
    },
    {
      "Classification": "spark-env",
      "Configurations": [
        {
          "Classification": "export",
          "Properties": {
            "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
          }
        }
       ],
       "Properties": {}
      },
      {
       "Classification": "spark-defaults",
       "Properties": {
         "spark.executorEnv.JAVA_HOME": "/usr/lib/jvm/java-11-amazon-
corretto.x86_64",
         "spark.sql.sources.fastS3PartitionDiscovery.enabled": "false"
       }
      }
  ]
```

7. Wählen Sie im Abschnitt Netzwerk eine Virtual Private Cloud (VPC) und ein Subnetz aus, die sich in Ihrem AWS Outposts Rack befinden. Weitere Informationen zu Amazon EMR in Outposts finden Sie unter EMR-Cluster in AWS Outposts im Verwaltungshandbuch für Amazon EMR.
8. Wählen Sie im Abschnitt EC2-Instance-Profil für Amazon EMR die IAM-Rolle aus, der die zuvor erstellte Berechtigungsrichtlinie angefügt ist.
9. Konfigurieren Sie Ihre verbleibenden Cluster-Einstellungen und wählen Sie dann Cluster erstellen aus.

Konfigurationsübersicht

In den folgenden Tabellen werden die S3A- und -SparkKonfigurationen sowie die Werte beschrieben, die für ihre Parameter angegeben werden müssen, wenn Sie einen Cluster einrichten, der S3 on Outposts mit Amazon EMR verwendet.

S3A-Konfigurationen

Parameter	Standardwert	Erforderlicher Wert für S3 on Outposts	Erklärung
`fs.s3a.aws.credentials.provider`	Wenn nicht angegeben, sucht S3A nach S3 im Regions-Bucket mit dem Namen des Outposts-Buckets.	Der Zugriffspunkt-ARN des S3-on-Outposts-Buckets	Amazon S3 on Outposts unterstützt reine Virtual-Private-Cloud(VPC)-Zugriffspunkte als einzige Möglichkeit, auf Ihre Outposts-Buckets zuzugreifen.
`fs.s3a.committer.name`	`file`	`magic`	Magic Committer ist der einzige unterstützte Committer für S3 on Outposts.
`fs.s3a.select.enabled`	`TRUE`	`FALSE`	S3 Select wird auf Outposts nicht unterstützt.
`JAVA_HOME`	`/usr/lib/jvm/java-8`	`/usr/lib/jvm/java-11-amazon-corretto.x86_64`	S3 on Outposts auf S3A erfordert Java Version 11.

Spark-Konfigurationen

Parameter	Standardwert	Erforderlicher Wert für S3 on Outposts	Erklärung
`spark.sql.sources.fastS3PartitionDiscovery.enabled`	`TRUE`	`FALSE`	S3 on Outposts unterstützt keine schnelle Partitionierung.

Parameter	Standardwert	Erforderlicher Wert für S3 on Outposts	Erklärung
`spark.executorEnv.JAVA_HOME`	`/usr/lib/jvm/java-8`	`/usr/lib/jvm/java-11-amazon-corretto.x86_64`	S3 on Outposts auf S3A erfordert Java-Version 11.

Überlegungen

Beachten Sie Folgendes, wenn Sie Amazon EMR in S3-on-Outposts-Buckets integrieren:

- Amazon S3 on Outposts wird mit Amazon-EMR-Version 7.0.0 und höher unterstützt.
- Der S3A-Konnektor ist erforderlich, um S3 on Outposts mit Amazon EMR zu verwenden. Nur S3A verfügt über die Funktionen, die für die Interaktion mit S3-on-Outposts-Buckets erforderlich sind. Informationen zur Einrichtung des S3A-Konnektors finden Sie unter Voraussetzungen.
- Amazon S3 on Outposts unterstützt nur die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) mit Amazon EMR. Weitere Informationen finden Sie unter the section called “Datenverschlüsselung”.
- Amazon S3 on Outposts unterstützt keine Schreibvorgänge mit dem S3A FileOutputCommitter. Schreibvorgänge mit den S3A FileOutputCommitter on S3 on Outposts-Buckets führen zu dem folgenden Fehler: InvalidStorageClass: Die von Ihnen angegebene Speicherklasse ist ungültig.
- Amazon S3 on Outposts wird mit Amazon EMR Serverless oder Amazon EMR in EKS nicht unterstützt.
- Amazon-EMR-Protokolle werden an einem von Ihnen ausgewählten regionalen Amazon S3-Speicherort gespeichert und nicht lokal im S3-on-Outposts-Bucket gespeichert.

Zwischenspeicherung von Autorisierung und Authentifizierung

S3 on Outposts speichert Authentifizierungs- und Autorisierungsdaten sicher lokal auf Outposts-Racks zwischen. Der Cache entfernt AWS-Region für jede Anforderung Roundtrips zum übergeordneten . Dadurch entfällt die Variabilität, die durch Netzwerk-Roundtrips eingeführt wird. Mit dem Authentifizierungs- und Autorisierungs-Cache in S3 on Outposts erhalten Sie konsistente Latenzen, die unabhängig von der Latenz der Verbindung zwischen den Outposts und der sind AWS-Region.

Wenn Sie eine S3-on-Outposts-API-Anfrage stellen, werden die Authentifizierungs- und Autorisierungsdaten sicher zwischengespeichert. Die zwischengespeicherten Daten werden dann verwendet, um nachfolgende S3-Objekt-API-Anforderungen zu authentifizieren. S3 on Outposts speichert Authentifizierungs- und Autorisierungsdaten nur zwischen, wenn die Anforderung mit Signature Version 4A (SigV4A) signiert wird. Der Cache wird lokal auf den Outposts innerhalb des S3-on-Outposts-Service gespeichert. Es wird asynchron aktualisiert, wenn Sie eine S3-API-Anfrage stellen. Der Cache ist verschlüsselt und es werden keine kryptografischen Klartextschlüssel auf Outposts gespeichert.

Der Cache ist bis zu 10 Minuten gültig, wenn der Outpost mit dem verbunden ist AWS-Region. Sie wird asynchron aktualisiert, wenn Sie eine S3-on-Outposts-API-Anforderung stellen, um sicherzustellen, dass die neuesten Richtlinien verwendet werden. Wenn der Outpost vom getrennt wird AWS-Region, ist der Cache bis zu 12 Stunden gültig.

Konfigurieren des Autorisierungs- und Authentifizierungs-Cache

S3 on Outposts speichert automatisch Authentifizierungs- und Autorisierungsdaten für Anfragen zwischen, die mit dem SigV4A-Algorithmus signiert sind. Weitere Informationen finden Sie unter Signieren von AWS API-Anforderungen im AWS Identity and Access Management - Benutzerhandbuch. Der SigV4A-Algorithmus ist in den neuesten Versionen der - AWS SDKs verfügbar. Sie können sie über eine Abhängigkeit von den AWS Common Runtime (CRT)-Bibliotheken abrufen.

Sie müssen die neueste Version des AWS SDK verwenden und die neueste Version des CRT installieren. Sie können beispielsweise ausführen, `pip install awscrt` um die neueste Version des CRT mit Boto3 zu erhalten.

S3 on Outposts speichert keine Authentifizierungs- und Autorisierungsdaten für Anfragen zwischen, die mit dem SigV4-Algorithmus signiert sind.

Validieren der SigV4A-Signatur

Sie können verwenden, AWS CloudTrail um zu überprüfen, ob Anfragen mit SigV4A signiert wurden. Weitere Informationen zum Einrichten von CloudTrail für S3 on Outposts finden Sie unter Überwachen von S3 in Outposts mit Protokollen in AWS CloudTrail.

Nachdem Sie konfiguriert haben CloudTrail, können Sie überprüfen, wie eine Anforderung im `SignatureVersion` Feld der CloudTrail Protokolle signiert wurde. Für Anforderungen, die mit SigV4A signiert wurden, ist auf `SignatureVersion` festgelegt`AWS 4-ECDSA-P256-SHA256`.

Anforderungen, die mit SigV4 signiert wurden, haben auf `SignatureVersion` gesetzt`AWS 4-HMAC-SHA256`.

Sicherheit in S3 on Outposts

Die Sicherheit in der Cloud hat bei AWS höchste Priorität. Als AWS-Kunde profitieren Sie von Rechenzentren und Netzwerkarchitekturen, die eingerichtet wurden, um die Anforderungen der anspruchsvollsten Organisationen in puncto Sicherheit zu erfüllen.

Sicherheit ist eine übergreifende Verantwortlichkeit zwischen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud selbst und Sicherheit in der Cloud:

- Sicherheit der Cloud selbst – AWS ist dafür verantwortlich, die Infrastruktur zu schützen, mit der AWS-Services in der AWS Cloud ausgeführt werden. AWS stellt Ihnen außerdem Services bereit, die Sie sicher nutzen können. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der AWS-Compliance-Programme regelmäßig. Informationen zu den Compliance-Programmen, die für Amazon S3 on Outposts gelten, finden Sie unter Im Rahmen des Compliance-Programms zugelassene AWS-Services.
- Sicherheit in der Cloud – Ihr Verantwortungsumfang wird durch den AWS-Service bestimmt, den Sie verwenden. Sie sind auch für andere Faktoren verantwortlich, einschließlich der Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation erläutert, wie das Modell der geteilten Verantwortung bei der Verwendung von S3 on Outposts zum Tragen kommt. Die folgenden Themen veranschaulichen, wie Sie S3 on Outposts zur Erfüllung Ihrer Sicherheits- und Compliance-Ziele konfigurieren können. Sie erfahren außerdem, wie Sie andere AWS-Services verwenden, um Ihre S3-on-Outposts-Ressourcen zu überwachen und zu schützen.

Themen

- Datenverschlüsselung in S3 on Outposts
- AWS PrivateLink für S3 on Outposts
- AWS-Signature Version 4 (SigV4) – Authentifizierungsspezifische Richtlinienschlüssel
- AWS-verwaltete Richtlinien für Amazon S3 on Outposts
- Verwenden von serviceverknüpften Rollen für S3 on Outposts

Datenverschlüsselung in S3 on Outposts

Standardmäßig werden alle in Amazon S3 on Outposts gespeicherten Daten mit serverseitiger Verschlüsselung über von Amazon S3 verwaltete Verschlüsselungsschlüssel (SSE-S3) verschlüsselt. Weitere Informationen finden Sie unter Verwenden serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3).

Sie können serverseitige Verschlüsselung optional mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) verwenden. Wenn Sie SSE-C verwenden möchten, geben Sie einen Verschlüsselungsschlüssel als Teil Ihrer Objekt-API-Anforderungen an. Die serverseitige Verschlüsselung verschlüsselt nur die Objektdaten, nicht die Metadaten des Objekts. Weitere Informationen finden Sie unter Verwenden von serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C).

> **Note**
>
> S3 on Outposts unterstützt keine serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS).

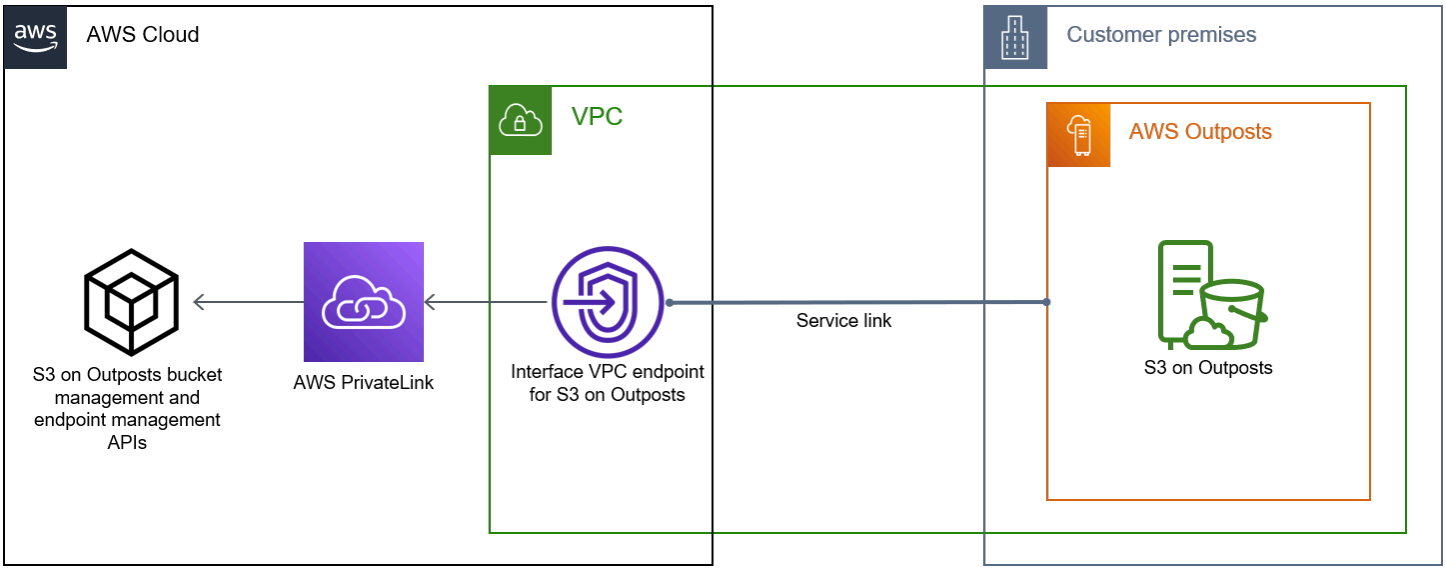
AWS PrivateLink für S3 on Outposts

S3 on Outposts unterstützt AWS PrivateLink, der über einen privaten Endpunkt in Ihrem VPN (Virtual Private Network) direkten Verwaltungszugriff auf Ihren S3-on-Outposts-Speicher bietet. Auf diese Weise können Sie Ihre interne Netzwerkarchitektur vereinfachen und Verwaltungsvorgänge auf Ihrem Outposts-Objektspeicher ausführen, indem Sie private IP-Adressen in Ihrer Virtual Private Cloud (VPC) verwenden. Die Verwendung von AWS PrivateLink macht die Nutzung öffentlicher IP-Adressen oder Proxyserver überflüssig.

Mit AWS PrivateLink für Amazon S3 on Outposts können Sie Schnittstellen-VPC-Endpunkte in Ihrer Virtual Private Cloud (VPC) bereitstellen, um auf Ihre S3-on-Outposts-APIs zur Bucket-Verwaltung und Endpunktverwaltung zuzugreifen. Schnittstellen-VPC-Endpunkte sind direkt von Anwendungen aus zugänglich, die in Ihrer VPC oder On-Premises über Ihr Virtual Private Network (VPN) oder AWS Direct Connect bereitgestellt sind. Sie können über AWS PrivateLink auf die APIs für die Bucket- und Endpunktverwaltung zugreifen. AWS PrivateLink unterstützt keine API-Operationen zur Datenübertragung wie GET, PUT und ähnliche APIs. Diese Vorgänge werden bereits privat über die Konfiguration des S3-on-Outposts-Endpunkts und des Zugriffspunkts übertragen. Weitere Informationen finden Sie unter Vernetzung für S3 on Outposts.

Schnittstellenendpunkte werden durch eine oder mehrere Elastic Network-Schnittstellen (ENIs) repräsentiert, denen private IP-Adressen aus Subnetzen in Ihrer VPC zugewiesen werden. Anfragen, die an Schnittstellenendpunkte für S3 on Outposts gestellt werden, werden automatisch an S3-on-Outposts-APIs zur Bucket- und Endpunktverwaltung im AWS-Netzwerk weitergeleitet. Sie können auch von On-Premises-Anwendungen in Ihrer VPC über AWS Direct Connect oder AWS Virtual Private Network (AWS VPN) auf Schnittstellen-Endpunkte zugreifen. Weitere Informationen darüber, wie Sie Ihre VPC mit Ihrem On-Premises-Netzwerk verbinden, finden Sie im AWS Direct Connect-Benutzerhandbuch und im AWS Site-to-Site VPN-Benutzerhandbuch.

Schnittstellenendpunkte leiten Anfragen für S3-on-Outposts-APIs für die Bucket- und Endpunktverwaltung über das AWS-Netzwerk und durch AWS PrivateLink weiter, wie im folgenden Diagramm veranschaulicht.



Allgemeine Informationen zu Schnittstellen-Endpunkten finden Sie unter VPC-Schnittstellen-Endpunkte (AWS PrivateLink) im AWS PrivateLink-Handbuch.

Themen

- Beschränkungen und Einschränkungen
- Zugriff auf S3-on-Outposts-Schnittstellenendpunkte
- Aktualisieren einer lokalen DNS-Konfiguration
- Erstellen eines VPC-Endpunkts für S3 on Outposts
- Erstellen von Bucket-Richtlinien und VPC-Endpunktrichtlinien für S3 on Outposts

Beschränkungen und Einschränkungen

Wenn Sie auf S3-on-Outposts-APIs für die Bucket- und Endpunktverwaltung über AWS PrivateLink zugreifen, gelten VPC-Einschränkungen. Weitere Informationen finden Sie unter Interface endpoint properties and limitations (Eigenschaften und Beschränkungen von Schnittstellen-Endpunkten) und AWS PrivateLink quotas (PrivateLink-Kontingente) im AWS PrivateLink-Leitfaden.

Darüber hinaus unterstützt AWS PrivateLink Folgendes nicht:

- Endpunkte für den Federal Information Processing Standard (FIPS).
- S3-on-Outposts-Datenübertragungs-APIs z. B. GET, PUT und ähnliche Objekt-API-Operationen.
- Privates DNS

Zugriff auf S3-on-Outposts-Schnittstellenendpunkte

Für den Zugriff auf S3-on-Outposts-APIs für die Bucket- und Endpunktverwaltung über AWS PrivateLink müssen Sie Ihre Anwendungen aktualisieren, damit diese endpunktspezifische DNS-Namen verwenden. Wenn Sie einen Schnittstellenendpunkt erstellen, generiert AWS PrivateLink zwei Arten von endpunktspezifischen S3-on-Outposts-Namen: regional und zonengebunden.

- Regionale DNS-Namen enthalten eine eindeutige VPC-Endpunkt-ID, eine Service-ID, die AWS-Region und `vpce.amazonaws.com`, z. B. *`vpce-1a2b3c4d-5e6f`*`.s3-outposts.us-east-1.vpce.amazonaws.com`.
- Zonengebundene DNS-Namen enthalten eine eindeutige VPC-Endpunkt-ID, die Availability Zone, eine Service-ID, die AWS-Region und `vpce.amazonaws.com`, z. B. *`vpce-1a2b3c4d-5e6f`***`-us-east-1a`**`.s3-outposts.us-east-1.vpce.amazonaws.com`. Sie können diese Option verwenden, wenn Ihre Architektur Availability Zones isoliert. Sie könnten zonengebundene Namen beispielsweise zur Fehlereingrenzung oder zur Senkung der regionalen Datenübertragungskosten verwenden.

> **Important**
>
> Die Endpunkte der S3-on-Outposts-Schnittstelle werden von der öffentlichen DNS-Domain aus aufgelöst. S3 on Outposts unterstützt kein privates DNS. Benutze den Parameter `--endpoint-url` für alle Bucket- und Endpunktverwaltungs-APIs.

Beispiele für AWS CLI

Verwenden Sie die Parameter `--region` und `--endpoint-url` für den Zugriff auf Bucket- und Endpunktverwaltungs-APIs über S3-on-Outposts-Schnittstellenendpunkte.

Example : Verwenden der Endpunkt-URL zum Auflisten von Buckets mit der S3-Steuerungs-API

Im folgenden Beispiel ersetzen Sie die Region *us-east-1*, die VPC-Endpunkt-URL *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* und die Konto-ID *111122223333* durch entsprechende Informationen.

```
aws s3control list-regional-buckets --region us-east-1 --endpoint-url
 https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com --account-
id 111122223333
```

AWS-SDK-Beispiele

Aktualisieren Sie Ihre SDKs auf die neueste Version und konfigurieren Sie Ihre Clients so, dass sie eine Endpunkt-URL für den Zugriff auf eine S3-Steuerungs-API für S3-on-Outposts-Schnittstellenendpunkte verwenden. Weitere Informationen finden Sie unter AWS-SDK-Beispiele für AWS PrivateLink.

SDK for Python (Boto3)

Example : Verwenden einer Endpunkt-URL, um auf die S3-Steuerungs-API zuzugreifen

Ersetzen Sie im folgenden Beispiel die Region *us-east-1* und die VPC-Endpunkt-URL *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com* durch entsprechende Informationen.

```
control_client = session.client(
service_name='s3control',
region_name='us-east-1',
endpoint_url='https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com'
)
```

Weitere Informationen finden Sie unter AWS PrivateLink für Amazon S3 im Boto3-Entwicklerhandbuch.

SDK for Java 2.x

Example : Verwenden einer Endpunkt-URL, um auf die S3-Steuerungs-API zuzugreifen

Ersetzen Sie im folgenden Beispiel die VPC-Endpunkt-URL *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com* und die Region *Region.US_EAST_1* durch entsprechende Informationen.

```
// control client
Region region = Region.US_EAST_1;
s3ControlClient = S3ControlClient.builder().region(region)

 .endpointOverride(URI.create("https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com"))
                                  .build()
```

Weitere Informationen finden Sie unter `S3ControlClient` in der AWS SDK for Java-API-Referenz.

Aktualisieren einer lokalen DNS-Konfiguration

Wenn Sie endpunktspezifische DNS-Namen für den Zugriff auf die Schnittstellenendpunkte für S3-on-Outposts-APIs für die Bucket- und Endpunktverwaltung verwenden, brauchen Sie Ihren On-Premises-DNS-Resolver nicht zu aktualisieren. Sie können den endpunktspezifischen DNS-Namen mit der privaten IP-Adresse des Schnittstellenendpunkts aus der öffentlichen S3-on-Outposts-DNS-Domäne auflösen.

Erstellen eines VPC-Endpunkts für S3 on Outposts

Informationen zum Erstellen eines VPC-Schnittstellenendpunkts für S3 on Outposts finden Sie unter Erstellen eines VPC-Endpunkts im AWS PrivateLink-Handbuch.

Erstellen von Bucket-Richtlinien und VPC-Endpunktrichtlinien für S3 on Outposts

Sie können eine Endpunktrichtlinie an Ihren VPC-Endpunkt anhängen, der den Zugriff auf S3 on Outposts steuert. Sie können auch die `aws:sourceVpce`-Bedingung in S3-on-Outposts-Bucket-Richtlinien verwenden, um den Zugriff auf bestimmte Buckets von einem bestimmten VPC-Endpunkt aus zu beschränken. Mit VPC-Endpunktrichtlinien können Sie den Zugriff auf S3-on-Outposts-APIs für die Bucket- und Endpunktverwaltung steuern. Mit Bucket-Richtlinien können Sie den Zugriff

auf S3-on-Outposts-APIs für die Bucket-Verwaltung steuern. Sie können jedoch den Zugriff auf Objektaktionen für S3 on Outposts nicht mit `aws:sourceVpce` verwalten.

Zugriffsrichtlinien für S3 on Outposts enthalten die folgenden Informationen:

- Der AWS Identity and Access Management (IAM)-Prinzipal, für den Aktionen erlaubt oder verweigert werden.
- Die S3-Steuerungsaktionen, die erlaubt oder verweigert werden.
- Die S3-on-Outposts-Ressourcen, für die Aktionen erlaubt oder verweigert werden.

Die folgenden Beispiele zeigen Richtlinien, die den Zugriff auf einen Bucket oder einen Endpunkt einschränken. Weitere Informationen über VPC-Konnektivität finden Sie unter Network-to-VPC connectivity options (Konnektivitätsoptionen vom Netzwerk zu VPC) im AWS-Whitepaper: Amazon Virtual Private Cloud Connectivity Options.

Important

- Wenn Sie die in diesem Abschnitt beschriebenen Beispielrichtlinien für VPC-Endpunkte anwenden, können Sie Ihren Zugriff auf den Bucket unbeabsichtigt blockieren. Bucket-Berechtigungen, die den Bucket-Zugriff auf Verbindungen beschränken, die von Ihrem VPC-Endpunkt ausgehen, können alle Verbindungen mit dem Bucket blockieren. Informationen zur Behebung dieses Problems finden Sie unter My bucket policy has the wrong VPC or VPC endpoint ID (Meine Bucket-Richtlinie hat die falsche VPC- oder VPC-Endpunkt-ID). Wie kann ich die Richtlinie so ändern, dass ich auf den Bucket zugreifen kann? im AWS Support Knowledge Center.
- Bevor Sie die folgende Bucket-Beispielrichtlinien verwenden, ersetzen Sie die VPC-Endpunkt-ID durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls können Sie nicht auf Ihren Bucket zugreifen.
- Wenn Ihre Richtlinie nur den Zugriff auf einen S3-on-Outposts-Bucket von einem bestimmten VPC-Endpunkt aus erlaubt, deaktiviert sie den Konsolenzugriff für diesen Bucket, da die Konsolenanforderungen nicht vom angegebenen VPC-Endpunkt stammen.

Themen

- Beispiel: Beschränken des Zugriffs auf einen bestimmten Bucket von einem VPC-Endpunkt aus

- Beispiel: Verweigern des Zugriffs von einem bestimmten VPC-Endpunkt aus in einer S3-on-Outposts-Bucket-Richtlinie

Beispiel: Beschränken des Zugriffs auf einen bestimmten Bucket von einem VPC-Endpunkt aus

Sie können eine Endpunktrichtlinie erstellen, die den Zugriff auf bestimmte S3-on-Outposts-Buckets beschränkt. Die folgende Richtlinie beschränkt den Zugriff für die GetBucketPolicy-Aktion nur auf *example-outpost-bucket*. Zum Verwenden dieses Beispiels ersetzen Sie die Beispielwerte durch Ihre eigenen.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909151",
  "Statement": [
    { "Sid": "Access-to-specific-bucket-only",
      "Principal": {"AWS":"111122223333"},
      "Action": "s3-outposts:GetBucketPolicy",
      "Effect": "Allow",
      "Resource": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-
bucket"
    }
  ]
}
```

Beispiel: Verweigern des Zugriffs von einem bestimmten VPC-Endpunkt aus in einer S3-on-Outposts-Bucket-Richtlinie

Die folgende S3-on-Outposts-Bucket-Richtlinie verweigert den Zugriff auf GetBucketPolicy im Bucket *example-outpost-bucket* über den *vpce-1a2b3c4d*-VPC-Endpunkt.

Die `aws:sourceVpce`-Bedingung gibt den Endpunkt an und erfordert keinen Amazon-Ressourcennamen (ARN) für die VPC-Endpunkt-Ressource, sondern nur die Endpunkt-ID. Zum Verwenden dieses Beispiels ersetzen Sie die Beispielwerte durch Ihre eigenen.

```
{
    "Version": "2012-10-17",
    "Id": "Policy1415115909152",
    "Statement": [
        {
```

```
            "Sid": "Deny-access-to-specific-VPCE",
            "Principal": {"AWS":"111122223333"},
            "Action": "s3-outposts:GetBucketPolicy",
            "Effect": "Deny",
            "Resource": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-
bucket",
            "Condition": {
                "StringEquals": {"aws:sourceVpce": "vpce-1a2b3c4d"}
            }
        }
    ]
}
```

AWS-Signature Version 4 (SigV4) – Authentifizierungsspezifische Richtlinienschlüssel

Die folgende Tabelle zeigt die Bedingungsschlüssel für die Authentifizierung mit AWS-Signature Version 4 (SigV4), die Sie mit Amazon S3 on Outposts verwenden können. In einer Bucket-Richtlinie können Sie diese Bedingungen hinzufügen, um ein bestimmtes Verhalten zu erzwingen, wenn Anforderungen mit der Signature Version 4 authentifiziert werden. Beispiele für Richtlinien finden Sie unter Beispiele für Bucket-Richtlinien, die mit der Signature Version 4 verbundene Bedingungsschlüssel verwenden. Weitere Informationen zur Authentifizierung von Anfragen mit Signature Version 4 finden Sie unter Authentifizieren von Anforderungen (AWS-Signature Version 4) in der Amazon Simple Storage Service API-Referenz

Anwendbare Schlüssel für **`s3-outposts:*`** Aktionen oder eine der S3 on Outposts-Aktionen

Anwendbare Schlüssel	Beschreibung
`s3-outposts:authType`	S3 on Outposts unterstützt verschiedene Methoden der Authentifizierung. Um eingehende Anfragen auf die Verwendung einer bestimmten Authentifizierungsmethode zu beschränken, können Sie diesen optionalen Bedingungsschlüssel verwenden. Sie können diesen Bedingungsschlüssel zum Beispiel verwenden, um nur den HTTP-`Authorization` -Header für die Authentifizierung von Anfragen zuzulassen. Zulässige Werte:

Anwendbare Schlüssel	Beschreibung
	`REST-HEADER` `REST-QUERY-STRING`
`s3-outposts:signatureAge`	Die Zeitspanne in Millisekunden, die eine Signatur in einer authentifizierten Anfrage gültig ist. Diese Bedingung gilt nur für vorsignierte URLs. In der Signature Version 4 ist der Signierschlüssel bis zu sieben Tage lang gültig. Daher sind die Signaturen auch bis zu sieben Tage lang gültig. Weitere Informationen finden Sie unter Einführung in das Signieren von Anfragen in der Amazon Simple Storage Service API-Referenz. Sie können diese Bedingung verwenden, um das Alter der Unterschrift weiter einzuschränken. Beispielwert: `600000`

Anwendbare Schlüssel	Beschreibung
`s3-outposts:x-amz-content-sha256`	Sie können diesen Bedingungsschlüssel verwenden, um nicht signierte Inhalte in Ihrem Bucket zu verbieten. Wenn Sie die Signature Version 4 verwenden, fügen Sie bei Anfragen, die den `Authorization` Header verwenden, den `x-amz-content-sha256` Header in die Signaturberechnung ein und setzen dann seinen Wert auf die Hash-Nutzlast. Sie können diesen Bedingungsschlüssel in Ihrer Bucket-Richtlinie verwenden, um alle Uploads zu verweigern, deren Nutzdaten nicht signiert sind. Zum Beispiel: • Verweigern Sie Uploads, die den `Authorization` -Header zur Authentifizierung von Anfragen verwenden, aber die Nutzdaten nicht signieren. Weitere Informationen finden Sie unter Übertragen von Nutzdaten in einem einzelnen Datenblock in der Amazon Simple Storage Service API-Referenz. • Verweigert Uploads, die vorsignierte URLs verwenden. Vorsignierte URLs haben immer eine `UNSIGNED_PAYLOAD` . Weitere Informationen finden Sie unter Authentifizierung von Anfragen und Authentifizierungsmethoden in der Amazon Simple Storage Service API-Referenz. Zulässiger Wert: `UNSIGNED-PAYLOAD`

Beispiele für Bucket-Richtlinien, die mit der Signature Version 4 verbundene Bedingungsschlüssel verwenden

Um die folgenden Beispiele zu verwenden, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen.

Example : **`s3-outposts:signatureAge`**

Die folgende Bucket-Richtlinie verweigert jede S3 on Outposts vorsignierte URL-Anfrage auf Objekte in `example-outpost-bucket`, wenn die Signatur mehr als 10 Minuten alt ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

Example : **`s3-outposts:authType`**

Die folgende Bucket-Richtlinie lässt nur Anfragen zu, die den `Authorization`-Header für die Anfrageauthentifizierung verwenden. Alle vorsignierten URL-Anfragen werden abgelehnt, da vorsignierte URLs Abfrageparameter verwenden, um Anfrage- und Authentifizierungsinformationen bereitzustellen. Weitere Informationen finden Sie unter Authentifizierungsmethoden in der Amazon Simple Storage Service API-Referenz.

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
               "Sid": "Allow only requests that use the Authorization header for
 request authentication. Deny presigned URL requests.",
               "Effect": "Deny",
               "Principal": {"AWS":"111122223333"},
               "Action": "s3-outposts:*",
               "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
               "Condition": {
                     "StringNotEquals": {
```

```
                        "s3-outposts:authType": "REST-HEADER"
                    }
                }
            }
    ]
}
```

Example : **`s3-outposts:x-amz-content-sha256`**

Die folgende Bucket-Richtlinie verweigert alle Uploads mit unsignierten Nutzdaten, z. B. Uploads, die vorsignierte URLs verwenden. Weitere Informationen finden Sie unter Authentifizierung von Anfragen und Authentifizierungsmethoden in der Amazon Simple Storage Service API-Referenz.

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
               "Sid": "Deny uploads with unsigned payloads.",
               "Effect": "Deny",
               "Principal": {"AWS":"111122223333"},
               "Action": "s3-outposts:*",
               "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
               "Condition": {
                     "StringEquals": {
                           "s3-outposts:x-amz-content-sha256": "UNSIGNED-PAYLOAD"
                     }
               }
         }
   ]
}
```

AWS-verwaltete Richtlinien für Amazon S3 on Outposts

Eine von AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von AWS erstellt und verwaltet wird. Von AWS verwaltete Richtlinien stellen Berechtigungen für viele häufige Anwendungsfälle bereit, damit Sie beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS-verwaltete Richtlinien möglicherweise nicht die geringsten Berechtigungen für Ihre spezifischen Anwendungsfälle gewähren, da sie für alle AWS-Kunden verfügbar sind.

Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie kundenverwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Die Berechtigungen, die in den von AWS verwalteten Richtlinien definiert sind, können nicht geändert werden. Wenn AWS Berechtigungen aktualisiert, die in einer von AWS verwalteten Richtlinie definiert werden, wirkt sich das Update auf alle Prinzipalidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWSaktualisiert am wahrscheinlichsten eine von AWS verwaltete Richtlinie, wenn ein neuer AWS-Service gestartet wird oder neue API-Operationen für bestehende Services verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS-verwaltete Richtlinie: AWSS3OnOutPostsServiceRolePolicy

Hilft Ihnen im Rahmen der serviceverknüpften Rolle `AWSServiceRoleForS3OnOutposts` bei der Verwaltung von Netzwerkressourcen.

Informationen zum Anzeigen der Berechtigungen für diese Richtlinie finden Sie unter AWSS3OnOutpostsServiceRolePolicy.

Änderungen von S3 on Outposts zu AWS-verwalteten Richtlinien

Sehen Sie sich Details über Aktualisierungen an von AWS verwalteten Richtlinien für S3 on Outposts an, seit der Service diese Änderungen nachverfolgt.

Änderung	Beschreibung	Datum
S3 on Outposts hat `AWSS3OnOutpostsServiceRolePolicy` hinzugefügt	S3 on Outposts hat `AWSS3OnOutpostsServiceRolePolicy` als Teil der serviceverknüpften Rolle `AWSServiceRoleForS3OnOutposts` hinzugefügt, die bei der Verwaltung von Netzwerkressourcen hilft.	03. Oktober 2023
S3 on Outposts hat mit der Verfolgung von Änderungen begonnen	S3 on Outposts hat mit der Verfolgung von Änderunge	03. Oktober 2023

Änderung	Beschreibung	Datum
	n für seine AWS-verwaltete Richtlinien begonnen.	

Verwenden von serviceverknüpften Rollen für S3 on Outposts

Amazon S3 on Outposts verwendet mit AWS Identity and Access Management (IAM) verknüpfte Servicerollen. Eine serviceverknüpfte Rolle ist ein spezieller Typ einer IAM-Rolle, die direkt mit S3 on Outposts verknüpft ist. Serviceverknüpfte Rollen werden von S3 on Outposts vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle vereinfacht das Einrichten von S3 on Outposts, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. S3 on Outposts definiert die Berechtigungen seiner serviceverknüpften Rollen. Sofern keine andere Konfiguration festgelegt wurde, kann nur S3 on Outposts die Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre S3-on-Outposts-Ressourcen, da Sie nicht versehentlich die Berechtigung für den Zugriff auf die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS-Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rollen angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Berechtigungen von serviceverknüpften Rollen für S3 on Outposts

S3 on Outposts verwendet die serviceverknüpfte Rolle AWSServiceRoleForS3OnOutposts, um Sie bei der Verwaltung von Netzwerkressourcen zu unterstützen.

Die serviceverknüpfte Rolle `AWSServiceRoleForS3OnOutposts` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `s3-outposts.amazonaws.com`

Die Rollenberechtigungsrichtlinie `AWSS3OnOutpostsServiceRolePolicy` ermöglicht S3 on Outposts die Durchführung der folgenden Aktionen für die angegebenen Ressourcen:

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcs",
                "ec2:DescribeCoipPools",
                "ec2:GetCoipPoolUsage",
                "ec2:DescribeAddresses",
                "ec2:DescribeLocalGatewayRouteTableVpcAssociations"
            ],
            "Resource": "*",
            "Sid": "DescribeVpcResources"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ],
            "Sid": "CreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
```

```
            },
            "Sid": "CreateTagsForCreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:ipv4pool-ec2/*"
            ],
            "Sid": "AllocateIpAddress"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:elastic-ip/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "CreateTagsForAllocateIpAddress"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteNetworkInterfacePermission",
                "ec2:DisassociateAddress",
                "ec2:ReleaseAddress",
                "ec2:AssociateAddress"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "S3 On Outposts"
                }
```

```
            },
            "Sid": "ReleaseVpcResources"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "CreateNetworkInterface",
                        "AllocateAddress"
                    ],
                    "aws:RequestTag/CreatedBy": [
                        "S3 On Outposts"
                    ]
                }
            },
            "Sid": "CreateTags"
        }
    ]
}
```

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. eine Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für S3 on Outposts

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie einen Endpunkt für S3 on Outposts in der AWS Management Console-, der AWS CLI- oder der AWS-API erstellen, erstellt S3 on Outposts die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie einen Endpunkt für S3 on Outposts erstellen, erstellt S3 on Outposts die serviceverknüpfte Rolle erneut für Sie.

Sie können auch die IAM-Konsole verwenden, um eine serviceverknüpfte Rolle mit dem Anwendungsfall S3 on Outposts zu erstellen. Erstellen Sie in der AWS CLI oder der AWS-API

eine servicegebundene Rolle mit dem Servicenamen `s3-outposts.amazonaws.com`. Weitere Informationen finden Sie unter Erstellen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch. Wenn Sie diese servicegebundene Rolle löschen, können Sie mit demselben Verfahren die Rolle erneut erstellen.

Bearbeiten einer serviceverknüpften Rolle für S3 on Outposts

S3 on Outposts verhindert die Bearbeitung der `AWSServiceRoleForS3OnOutposts` serviceverknüpften Rolle. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach der Erstellung nicht berücksichtigt werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für S3 on Outposts

Wenn Sie ein Feature oder einen Service, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch die Ressourcen für Ihre serviceverknüpften Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

> **Note**
>
> Wenn der S3-on-Outposts-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt das Löschen möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

So löschen Sie Ressourcen von S3 on Outposts, die von der Rolle „AWSServiceRoleForS3OnOutposts“ verwendet werden

1. Löschen Sie die S3-on-Outposts-Endpunkte in Ihrem AWS-Konto in allen AWS-Regionen.
2. Löschen Sie die serviceverknüpfte Rolle mit IAM.

   Verwenden Sie die IAM-Konsole, AWS CLI- oder AWS-API, um die `AWSServiceRoleForS3OnOutposts` serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte S3-on-Outposts-Rollen

S3 on Outposts unterstützt die Verwendung von serviceverknüpften Rollen in allen AWS-Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter S3-on-Outposts-Regionen und -Endpunkte.

Verwaltung von S3-on-Outposts-Speicher

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (`OUTPOSTS`). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Weitere Informationen zum Verwalten und Freigeben Ihrer Speicherkapazität von Amazon S3 in Outposts finden Sie in den folgenden Themen.

Themen

- Verwalten der S3-Versionierung für Ihren S3-on-Outposts-Bucket
- Erstellen und Verwalten einer Lebenszyklus-Konfiguration für Ihren Amazon-S3-on-Outposts-Bucket
- Replikation von Objekten für S3 in Outposts
- Freigabe von S3 in Outposts mithilfe von AWS RAM
- Sonstige AWS-Services, die S3 on Outposts verwenden

Verwalten der S3-Versionierung für Ihren S3-on-Outposts-Bucket

Wenn diese Option aktiviert ist, speichert die S3-Versionsverwaltung mehrere unterschiedliche Kopien eines Objekts im selben Bucket. Sie können die S3-Versionsverwaltung verwenden, um sämtliche Versionen aller Objekte in Ihren Outposts-Buckets zu speichern, abzurufen

oder wiederherzustellen. Mit der S3-Versionsverwaltung können Sie Versionen sowohl nach unbeabsichtigten Benutzeraktionen als auch nach Anwendungsfehlern problemlos wiederherstellen.

Buckets von Amazon S3 on Outposts verfügen über drei Versionsverwaltungsstatus:

- Unversioned (Nicht versioniert) – Wenn Sie die S3-Versionsverwaltung für Ihren Bucket noch nie aktiviert oder ausgesetzt haben, ist er nicht versioniert und gibt keinen S3-Versionsverwaltungsstatus zurück. Weitere Informationen über das S3-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.
- Enabled (Aktiviert) – Aktiviert die S3-Versionsverwaltung für die Objekte im Bucket. Alle Objekte, die dem Bucket hinzugefügt werden, erhalten eine eindeutige Versions-ID. Objekte, die zum Zeitpunkt der Aktivierung des Versioning bereits im Bucket vorhanden waren, haben die Versions-ID`null` . Wenn Sie diese (oder andere) Objekte mit anderen Operationen wie PutObject verändern, erhalten die neuen Objekte eine eindeutige Versions-ID.
- Suspended (Ausgesetzt) – Setzt die S3-Versionsverwaltung für die Objekte im Bucket aus. Alle Objekte, die dem Bucket hinzugefügt werden, nachdem die Versionsverwaltung ausgesetzt wurde, erhalten die Versions-ID `null`. Weitere Informationen finden Sie unter Hinzufügen von Objekten zu Buckets mit ausgesetztem Versioning.

Nachdem Sie die S3-Versionsverwaltung für einen S3-on-Outposts-Bucket aktiviert haben, kann er nicht mehr auf einen nicht versionierten Status zurückgesetzt werden. Sie können die Versionsverwaltung jedoch aussetzen. Weitere Informationen über das S3-Versioning finden Sie unter Verwenden der Versioning in S3-Buckets.

Sie haben für jedes Objekt in Ihrem Bucket eine aktuelle Version und keine oder mehr vorherige Versionen. Damit die Speicherkosten gesenkt werden, können Sie die S3-Lebenszyklusregeln für Ihren Bucket so konfigurieren, dass vorherige Versionen nach einem bestimmten Zeitraum ablaufen. Weitere Informationen finden Sie unter Erstellen und Verwalten einer Lebenszyklus-Konfiguration für Ihren Amazon-S3-on-Outposts-Bucket.

Die folgenden Beispiele veranschaulichen, wie Sie die Versionsverwaltung für einen vorhandenen S3-on-Outposts-Bucket mithilfe der AWS Management Console und der AWS Command Line Interface (AWS CLI) aktivieren oder aussetzen. Informationen zum Erstellen eines Buckets mit aktivierter S3-Versionsverwaltung finden Sie unter Erstellen eines S3-on-Outposts-Buckets.

> **Note**
>
> Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das ihm Aktionen zuweisen kann. Buckets verfügen über Konfigurationseigenschaften wie Outpost, Tags, Standard-Verschlüsselung und Zugriffspunkteinstellungen. Zu den Zugriffspunkteinstellungen gehören die Virtual Private Cloud (VPC), die Zugriffspunkt-Richtlinie für den Zugriff auf die Objekte im Bucket sowie andere Metadaten. Weitere Informationen finden Sie unter Spezifikationen für S3 auf Outposts.

Verwenden der S3-Konsole

So bearbeiten Sie die S3-Versionsverwaltungseinstellungen für Ihren Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, für den Sie die S3-Versionsverwaltung aktivieren möchten.
4. Wählen Sie die Registerkarte Properties (Eigenschaften) aus.
5. Wählen Sie unter Bucket Versioning (Bucket-Versioning) die Option Edit (Bearbeiten).
6. Bearbeiten Sie die S3-Versionsverwaltungseinstellung für den Bucket, indem Sie eine der folgenden Optionen auswählen:
   - Wenn Sie die S3-Versionsverwaltung aussetzen und die Erstellung neuer Objektversionen anhalten möchten, wählen Sie Suspend (Aussetzen) aus.
   - Möchten Sie die S3-Versionsverwaltung aktivieren und mehrere unterschiedliche Kopien jedes Objekts speichern, wählen Sie Enable (Aktivieren) aus.
7. Wählen Sie Save Changes (Änderungen speichern).

Verwendung der AWS CLI

Wenn Sie die S3-Versionsverwaltung für Ihren Bucket mithilfe der AWS CLI aktivieren oder aussetzen möchten, verwenden Sie den Befehl `put-bucket-versioning`, wie in den folgenden Beispielen gezeigt. Wenn Sie diese Beispiele verwenden möchten, ersetzen Sie jeden *`user input placeholder`* durch Ihre Informationen.

Weitere Informationen finden Sie unter put-bucket-versioning in der AWS CLI-Referenz.

Example : S3-Versionsverwaltung aktivieren

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --versioning-configuration Status=Enabled
```

Example : S3-Versionsverwaltung aussetzen

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --versioning-configuration Status=Suspended
```

Erstellen und Verwalten einer Lebenszyklus-Konfiguration für Ihren Amazon-S3-on-Outposts-Bucket

Sie können S3 Lifecycle verwenden, um die Speicherkapazität für Amazon S3 on Outposts zu optimieren. Sie können Lebenszyklusregeln erstellen, um Objekte ablaufen zu lassen, wenn sie veralten oder durch neuere Versionen ersetzt werden. Sie können eine Lebenszyklusregel erstellen, aktivieren, deaktivieren oder löschen.

Weitere Informationen zum S3-Lebenszyklus finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

> **Note**
>
> Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das eine Lebenszyklusregel erstellen, aktivieren, deaktivieren oder löschen kann.

Informationen zum Erstellen und Verwalten der Lebenszyklus-Konfiguration für Ihren S3-on-Outposts-Bucket finden Sie in den folgenden Themen.

Themen

- Erstellen und Verwalten einer Lebenszyklusregel mithilfe der AWS Management Console
- Erstellen und Verwalten einer Lebenszyklus-Konfiguration mithilfe der AWS CLI und dem SDK for Java

Erstellen und Verwalten einer Lebenszyklusregel mithilfe der AWS Management Console

Sie können S3 Lifecycle verwenden, um die Speicherkapazität für Amazon S3 on Outposts zu optimieren. Sie können Lebenszyklusregeln erstellen, um Objekte ablaufen zu lassen, wenn sie veralten oder durch neuere Versionen ersetzt werden. Sie können eine Lebenszyklusregel erstellen, aktivieren, deaktivieren oder löschen.

Weitere Informationen zum S3-Lebenszyklus finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

> **Note**
>
> Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das eine Lebenszyklusregel erstellen, aktivieren, deaktivieren oder löschen kann.

Weitere Informationen zum Erstellen und Verwalten einer Lebenszyklusregel für S3 on Outposts mithilfe der AWS Management Console finden Sie in den folgenden Themen.

Themen

- Erstellen einer Lebenszyklusregel
- Aktivieren einer Lebenszyklusregel
- Bearbeiten einer Lebenszyklusregel
- Löschen einer Lebenszyklusregel

Erstellen einer Lebenszyklusregel

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, für den Sie eine Lebenszyklusregel erstellen möchten.
4. Wählen Sie die Registerkarte Management (Verwaltung) und dann die Option Create lifecycle rule (Lebenszyklusregel erstellen) aus.
5. Geben Sie einen Wert für Lifecycle rule name (Lebenszyklusregelname) ein.
6. Wählen Sie unter Rule scope (Regelumfang) eine der folgenden Optionen aus:

- Wenn Sie den Umfang mit bestimmten Filtern einschränken möchten, wählen Sie Limit the scope of this rule using one or more filters (Geltungsbereich dieser Regel mit einem oder mehreren Filtern einschränken) aus. Fügen Sie anschließend einen Präfixfilter, Tags oder eine Objektgröße hinzu.
- Wenn Sie diese Lebenszyklusregel auf alle Objekte im Bucket anwenden möchten, wählen Sie Apply to all objects in the bucket (Auf alle Objekte im Bucket anwenden) aus.

7. Wählen Sie unter Lifecycle rule actions (Lebenszyklusregelaktionen) eine der folgenden Optionen aus:

   - Expire current versions of objects (Aktuelle Objektversionen ablaufen lassen) – Bei Buckets mit aktivierter Versionsverwaltung fügt S3 on Outposts eine Löschmarkierung hinzu und behält die Objekte als nicht aktuelle Versionen bei. Bei Buckets, die keine S3-Versionsverwaltung verwenden, löscht S3 on Outposts die Objekte dauerhaft.
   - Permanently delete noncurrent versions of objects (Vorherige Objektversionen dauerhaft löschen) – S3 on Outposts löscht nicht aktuelle Objektversionen dauerhaft.
   - Delete expired object delete markers or incomplete multipart uploads (Abgelaufene Objektlöschmarkierungen oder unvollständige mehrteilige Uploads löschen) – S3 on Outposts löscht Löschmarkierungen für abgelaufene Objekte oder unvollständige mehrteilige Uploads dauerhaft.

     Wenn Sie den Umfang Ihrer Lebenszyklusregel mithilfe von Objekt-Tags einschränken, können Sie die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) nicht auswählen. Sie können die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) auch nicht auswählen, wenn Sie Expire current object versions (Aktuelle Objektversionen ablaufen lassen) aktiviert haben.

   > **Note**
   >
   > Größenabhängige Filter können nicht mit Löschmarkierungen und unvollständigen mehrteiligen Uploads verwendet werden.

8. Wenn Sie Expire current versions of objects (Aktuelle Objektversionen ablaufen lassen) oder Permanently delete noncurrent versions of objects (Vorherige Objektversionen dauerhaft

löschen) ausgewählt haben, konfigurieren Sie den Regelauslöser basierend auf einem bestimmten Datum oder dem Alter des Objekts.

9. Wenn Sie die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) ausgewählt haben, wählen Sie zur Bestätigung dieses Vorgangs die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) erneut aus.
10. Überprüfen Sie unter Timeline Summary (Timeline-Zusammenfassung) Ihre Lebenszyklusregel und wählen Sie Create rule (Regel erstellen) aus.

Aktivieren einer Lebenszyklusregel

So aktivieren oder deaktivieren Sie eine Bucket-Lebenszyklusregel

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, für den Sie eine Lebenszyklusregel deaktivieren möchten.
4. Wählen Sie die Registerkarte Management (Verwaltung) und dann unter Lifecycle rule (Lebenszyklusregel) die Regel aus, die Sie aktivieren oder deaktivieren möchten.
5. Wählen Sie für Aktion die Option Regel aktivieren oder deaktivieren aus.

Bearbeiten einer Lebenszyklusregel

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, für den Sie eine Lebenszyklusregel bearbeiten möchten.
4. Wählen Sie die Registerkarte Verwaltung und wählen Sie die Lebenszyklusregel aus, die Sie bearbeiten möchten.
5. (Optional) Aktualisieren Sie den Wert für Lifecycle rule name (Lebenszyklusregelname).
6. Bearbeiten Sie unter Rule scope (Regelumfang) den Umfang nach Bedarf:

   - Wenn Sie den Umfang mit bestimmten Filtern einschränken möchten, wählen Sie Limit the scope of this rule using one or more filters (Geltungsbereich dieser Regel mit einem oder mehreren Filtern einschränken) aus. Fügen Sie anschließend einen Präfixfilter, Tags oder eine Objektgröße hinzu.

- Wenn Sie diese Lebenszyklusregel auf alle Objekte im Bucket anwenden möchten, wählen Sie Apply to all objects in the bucket (Auf alle Objekte im Bucket anwenden) aus.

7. Wählen Sie unter Lifecycle rule actions (Lebenszyklusregelaktionen) eine der folgenden Optionen aus:

   - Expire current versions of objects (Aktuelle Objektversionen ablaufen lassen) – Bei Buckets mit aktivierter Versionsverwaltung fügt S3 on Outposts eine Löschmarkierung hinzu und behält die Objekte als nicht aktuelle Versionen bei. Bei Buckets, die keine S3-Versionsverwaltung verwenden, löscht S3 on Outposts die Objekte dauerhaft.
   - Permanently delete noncurrent versions of objects (Vorherige Objektversionen dauerhaft löschen) – S3 on Outposts löscht nicht aktuelle Objektversionen dauerhaft.
   - Delete expired object delete markers or incomplete multipart uploads (Abgelaufene Objektlöschmarkierungen oder unvollständige mehrteilige Uploads löschen) – S3 on Outposts löscht Löschmarkierungen für abgelaufene Objekte oder unvollständige mehrteilige Uploads dauerhaft.

     Wenn Sie den Umfang Ihrer Lebenszyklusregel mithilfe von Objekt-Tags einschränken, können Sie die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) nicht auswählen. Sie können die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) auch nicht auswählen, wenn Sie Expire current object versions (Aktuelle Objektversionen ablaufen lassen) aktiviert haben.

   > **Note**
   >
   > Größenabhängige Filter können nicht mit Löschmarkierungen und unvollständigen mehrteiligen Uploads verwendet werden.

8. Wenn Sie Expire current versions of objects (Aktuelle Objektversionen ablaufen lassen) oder Permanently delete noncurrent versions of objects (Vorherige Objektversionen dauerhaft löschen) ausgewählt haben, konfigurieren Sie den Regelauslöser basierend auf einem bestimmten Datum oder dem Objektalter.
9. Wenn Sie die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) ausgewählt haben, wählen Sie zur Bestätigung dieses Vorgangs die Option Delete expired object delete markers (Löschmarkierungen für abgelaufenes Objekt löschen) erneut aus.

10. Wählen Sie Save (Speichern).

Löschen einer Lebenszyklusregel

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Outposts-Bucket aus, für den Sie eine Lebenszyklusregel löschen möchten.
4. Wählen Sie die Registerkarte Management (Verwaltung) und unter Lifecycle rule (Lebenszyklusregel) die Regel aus, die Sie löschen möchten.
5. Wählen Sie Delete (Löschen).

Erstellen und Verwalten einer Lebenszyklus-Konfiguration mithilfe der AWS CLI und dem SDK for Java

Sie können S3 Lifecycle verwenden, um die Speicherkapazität für Amazon S3 on Outposts zu optimieren. Sie können Lebenszyklusregeln erstellen, um Objekte ablaufen zu lassen, wenn sie veralten oder durch neuere Versionen ersetzt werden. Sie können eine Lebenszyklusregel erstellen, aktivieren, deaktivieren oder löschen.

Weitere Informationen zum S3-Lebenszyklus finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

> **Note**
>
> Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das eine Lebenszyklusregel erstellen, aktivieren, deaktivieren oder löschen kann.

Weitere Informationen zum Erstellen und Verwalten einer Lebenszyklus-Konfiguration für einen S3-on-Outposts-Bucket mithilfe der AWS Command Line Interface (AWS CLI) und von AWS SDK for Java finden Sie in den folgenden Beispielen.

Themen

- PUT-Befehl für eine Lebenszyklus-Konfiguration
- GET-Befehl für eine Lebenszyklus-Konfiguration für einen S3-on-Outposts-Bucket

PUT-Befehl für eine Lebenszyklus-Konfiguration

AWS CLI

Im folgenden AWS CLI-Beispiel wird eine Lebenszyklus-Konfigurationsrichtlinie in einen Outposts-Bucket eingefügt. Diese Richtlinie gibt an, dass alle Objekte mit dem gekennzeichneten Präfix (*myprefix*) und Tags nach 10 Tagen ablaufen. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

1. Speichern Sie die Richtlinie für die Lebenszyklus-Konfiguration in einer JSON-Datei. In diesem Beispiel heißt die Datei `lifecycle1.json`.

```
{
    "Rules": [
        {
            "ID": "id-1",
            "Filter": {
                "And": {
                    "Prefix": "myprefix",
                    "Tags": [
                        {
                            "Value": "mytagvalue1",
                            "Key": "mytagkey1"
                        },
                        {
                            "Value": "mytagvalue2",
                            "Key": "mytagkey2"
                        }
                    ],
                    "ObjectSizeGreaterThan": 1000,
                    "ObjectSizeLessThan": 5000
                }
            },
            "Status": "Enabled",
            "Expiration": {
                "Days": 10
            }
        }
    ]
}
```

2. Senden Sie die JSON-Datei als Teil des CLI-Befehls `put-bucket-lifecycle-configuration`. Zum Verwenden dieses Befehls ersetzen Sie *user input*

placeholder durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter put-bucket-lifecycle-configuration in der AWS CLI-Referenz.

```
aws s3control put-bucket-lifecycle-configuration --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket --lifecycle-configuration file://lifecycle1.json
```

SDK for Java

Im folgenden SDK-für-Java-Beispiel wird eine Lebenszyklus-Konfiguration in einen Outposts-Bucket eingefügt. Diese Lebenszykluskonfiguration gibt an, dass alle Objekte mit dem gekennzeichneten Präfix (*myprefix*) und Tags nach 10 Tagen ablaufen. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie jeden *user input placeholder* durch Ihre Informationen. Weitere Informationen finden Sie unter PutBucketLifecycleConfiguration in der API-Referenz zu Amazon Simple Storage Service.

```
import com.amazonaws.services.s3control.model.*;

public void putBucketLifecycleConfiguration(String bucketArn) {

    S3Tag tag1 = new S3Tag().withKey("mytagkey1").withValue("mytagkey1");
    S3Tag tag2 = new S3Tag().withKey("mytagkey2").withValue("mytagkey2");

    LifecycleRuleFilter lifecycleRuleFilter = new LifecycleRuleFilter()
            .withAnd(new LifecycleRuleAndOperator()
                    .withPrefix("myprefix")
                    .withTags(tag1, tag2))
                    .withObjectSizeGreaterThan(1000)
                    .withObjectSizeLessThan(5000);

    LifecycleExpiration lifecycleExpiration = new LifecycleExpiration()
            .withExpiredObjectDeleteMarker(false)
            .withDays(10);

    LifecycleRule lifecycleRule = new LifecycleRule()
            .withStatus("Enabled")
            .withFilter(lifecycleRuleFilter)
            .withExpiration(lifecycleExpiration)
            .withID("id-1");

    LifecycleConfiguration lifecycleConfiguration = new LifecycleConfiguration()
```

```
            .withRules(lifecycleRule);

    PutBucketLifecycleConfigurationRequest reqPutBucketLifecycle = new
 PutBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withLifecycleConfiguration(lifecycleConfiguration);

    PutBucketLifecycleConfigurationResult respPutBucketLifecycle =
 s3ControlClient.putBucketLifecycleConfiguration(reqPutBucketLifecycle);
    System.out.printf("PutBucketLifecycleConfiguration Response: %s%n",
 respPutBucketLifecycle.toString());

}
```

GET-Befehl für eine Lebenszyklus-Konfiguration für einen S3-on-Outposts-Bucket

AWS CLI

Im folgenden Beispiel AWS CLI wird eine Lebenszykluskonfiguration für einen Outposts-Bucket abgerufen. Zum Verwenden dieses Befehls ersetzen Sie *user input placeholder* durch Ihre eigenen Informationen. Weitere Informationen über diesen Befehl finden Sie unter get-bucket-lifecycle-configuration in der AWS CLI-Referenz.

```
aws s3control get-bucket-lifecycle-configuration --account-id 123456789012 --bucket
 arn:aws:s3-outposts:<your-region>:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket
```

SDK for Java

Das folgende SDK für Java-Beispiel ruft eine Lebenszykluskonfiguration für einen Outposts-Bucket ab. Weitere Informationen finden Sie unter GetBucketLifecycleConfiguration in der API-Referenz zu Amazon Simple Storage Service.

```
import com.amazonaws.services.s3control.model.*;

public void getBucketLifecycleConfiguration(String bucketArn) {

    GetBucketLifecycleConfigurationRequest reqGetBucketLifecycle = new
 GetBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
```

```
            .withBucket(bucketArn);

    GetBucketLifecycleConfigurationResult respGetBucketLifecycle =
 s3ControlClient.getBucketLifecycleConfiguration(reqGetBucketLifecycle);
    System.out.printf("GetBucketLifecycleConfiguration Response: %s%n",
 respGetBucketLifecycle.toString());

}
```

Replikation von Objekten für S3 in Outposts

Mit S3 Replication in AWS Outposts können Sie Amazon S3 in Outposts so konfigurieren, dass S3-Objekte automatisch über verschiedene Outposts hinweg oder zwischen Buckets in demselben Outpost repliziert werden. Sie können S3 Replication in Outposts verwenden, um mehrere Replikate Ihrer Daten in denselben oder verschiedenen Outposts oder über verschiedene Konten hinweg zu verwalten und die Anforderungen an die Datenspeicherorte zu erfüllen. S3 Replication in Outposts hilft Ihnen dabei, Ihre Anforderungen an konforme Speicher und die Datenfreigabe zwischen verschiedenen Konten zu erfüllen. Wenn Sie sicherstellen müssen, dass Ihre Replikate mit den Quelldaten übereinstimmen, können Sie mit S3 Replication in Outposts Replikate Ihrer Objekte erstellen, die alle Metadaten enthalten, z. B. die Erstellungszeit des ursprünglichen Objekts, Tags und Versions-IDs.

S3 Replication in Outposts stellt außerdem detaillierte Metriken und Benachrichtigungen zum Überwachen des Status der Objektreplikation zwischen Buckets bereit. Sie können Amazon CloudWatch verwenden, um den Replikationsfortschritt zu überwachen, indem Sie die Bytes mit ausstehender Replikation, die Operationen mit ausstehender Replikation und die Replikationslatenz zwischen Ihren Quell- und Ziel-Buckets nachverfolgen. Um Konfigurationsprobleme schnell zu diagnostizieren und zu beheben, können Sie außerdem Amazon EventBridge so einrichten, dass Benachrichtigungen über Fehler bei Replikationsobjekten empfangen werden. Weitere Informationen hierzu finden Sie unter Verwalten Ihrer Replikation.

Themen

- Replikations-Konfiguration
- Anforderungen für S3 Replication in Outposts
- Was wird repliziert?
- Was wird nicht repliziert?
- Was wird von S3 Replication in Outposts nicht unterstützt?

- Einrichten der Replikation
- Verwalten Ihrer Replikation

Replikations-Konfiguration

S3 in Outposts speichert Replikations-Konfigurationen als XML. In der XML-Datei mit der Replikations-Konfiguration legen Sie eine AWS Identity and Access Management (IAM)-Rolle und mindestens eine Regel fest.

```
<ReplicationConfiguration>
    <Role>IAM-role-ARN</Role>
    <Rule>
        ...
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
</ReplicationConfiguration>
```

S3 in Outposts kann Objekte nur dann replizieren, wenn Sie die entsprechende Berechtigung erteilen. Sie erteilen S3 in Outposts Berechtigungen mit der IAM-Rolle, die Sie in der Replikations-Konfiguration angeben. S3 in Outposts übernimmt diese IAM-Rolle, um Objekte in Ihrem Namen zu replizieren. Sie müssen der IAM-Rolle die erforderlichen Berechtigungen erteilen, bevor Sie die Replikation starten. Weitere Informationen zu diesen Berechtigungen für S3 in Outposts finden Sie unter Erstellen einer IAM-Rolle.

In den folgenden Szenarien fügen Sie eine Regel zur Replikationskonfiguration hinzu:

- Sie möchten alle Objekte replizieren.
- Sie möchten eine Teilmenge der Objekte replizieren. Sie identifizieren die Teilmenge der Objekte, indem Sie einen Filter zur Regel hinzufügen. In dem Filter geben Sie ein Objektschlüsselpräfix, Markierungen oder eine Kombination aus beidem an, um die Objektteilmenge zu identifizieren, für die die Regel gilt.

Sie fügen mehrere Regeln zu einer Replikationskonfiguration hinzu, wenn Sie eine andere Teilmenge von Objekten replizieren möchten. In jeder Regel geben Sie einen Filter an, der eine andere Teilmenge von Objekten auswählt. Beispiel: Sie möchten Objekte mit dem Schlüsselpräfix `tax/` oder

`document/` replizieren. Dazu fügen Sie zwei Regeln hinzu, eine, die den `tax/`-Schlüsselpräfix-Filter angibt und eine andere, die das `document/`-Schlüsselpräfix angibt.

Weitere Informationen zur Replikations-Konfiguration und zu den Replikationsregeln von S3 in Outposts finden Sie unter ReplicationConfiguration in der API-Referenz zu Amazon Simple Storage Service.

Anforderungen für S3 Replication in Outposts

Für die Replikation ist Folgendes erforderlich:

- Der Outpost-CIDR-Zielbereich muss Ihrer Outpost-Quellsubnetztabelle zugeordnet sein. Weitere Informationen finden Sie unter Voraussetzungen für das Erstellen von Konfigurationsregeln.
- Für Quell- und Ziel-Buckets muss die S3-Versionsverwaltung aktiviert sein. Weitere Informationen über das Versioning finden Sie unter Verwalten der S3-Versionierung für Ihren S3-on-Outposts-Bucket.
- Amazon S3 in Outposts muss über die Berechtigung verfügen, Objekte aus dem Quell-Bucket in Ihrem Namen in den Ziel-Bucket zu replizieren. Dies bedeutet, dass Sie eine Servicerolle zum Delegieren von `GET`- und `PUT`-Berechtigungen an S3 in Outposts erstellen müssen.
  1. Bevor Sie die Servicerolle erstellen, benötigen Sie die `GET`-Berechtigung für den Quell-Bucket und die `PUT`-Berechtigung für den Ziel-Bucket.
  2. Um die Servicerolle zum Delegieren von Berechtigungen an S3 in Outposts erstellen zu können, müssen Sie zunächst Berechtigungen konfigurieren, damit eine IAM-Entität (ein Benutzer oder eine Rolle) die Aktionen `iam:CreateRole` und `iam:PassRole` ausführen kann. Anschließend erlauben Sie der IAM-Entität, die Servicerolle zu erstellen. Damit S3 in Outposts die Servicerolle in Ihrem Namen annehmen kann und um `GET`- und `PUT`-Berechtigungen an S3 in Outposts zu delegieren, müssen Sie der Rolle die erforderlichen Vertrauens- und Berechtigungsrichtlinien zuordnen. Weitere Informationen zu diesen Berechtigungen für S3 in Outposts finden Sie unter Erstellen einer IAM-Rolle. Weitere Informationen zum Erstellen einer Servicerolle finden Sie unter Erstellen einer Servicerolle.

Was wird repliziert?

Standardmäßig repliziert S3 in Outposts Folgendes:

- Objekte, die nach dem Hinzufügen einer Replikations-Konfiguration erstellt wurden.

- Objektmetadaten von den Quellobjekten zu den Replikaten Informationen zum Replizieren von Metadaten aus den Replikaten zu den Quellobjekten finden Sie unter Replikationsstatus, wenn die Synchronisierung von Amazon-S3-Replikatänderungen in Outposts aktiviert ist.
- Objekt-Markierungen, sofern vorhanden.

Auswirkungen von LöschVorgänge auf die Replikation

Wenn Sie ein Objekt aus dem Quell-Bucket löschen, werden standardmäßig die folgenden Aktionen ausgeführt:

- Wenn Sie eine `DELETE`-Anforderung ohne Angabe einer Objektversions-ID stellen, fügt S3 in Outposts eine Löschmarkierung hinzu. S3 in Outposts geht wie folgt mit der Löschmarkierung um:
  - S3 in Outposts repliziert die Löschmarkierung standardmäßig nicht.
  - Sie können jedoch die Löschmarkierungs-Replikation zu nicht Tag-basierten Regeln hinzufügen. Weitere Informationen zum Aktivieren der Löschmarkierungs-Replikation in Ihrer Replikations-Konfiguration finden Sie unter Verwenden der S3-Konsole.
- Wenn Sie in einer `DELETE`-Anforderung eine zu löschende Objektversions-ID angeben, löscht S3 in Outposts diese Objektversion im Quell-Bucket dauerhaft. Die Löschung wird jedoch nicht in den Ziel-Buckets repliziert. Anders ausgedrückt: Dieselbe Objektversion wird aus den Ziel-Buckets nicht gelöscht. Dieses Verhalten schützt Daten vor böswilligen Löschungen.

Was wird nicht repliziert?

Standardmäßig repliziert S3 in Outposts Folgendes nicht:

- Objekte im Quell-Bucket, bei denen es sich um Replikate handelt, die von einer anderen Replikationsregel erstellt wurden. Zum Beispiel: Angenommen Sie konfigurieren eine Replikation, bei der Bucket A die Quelle und Bucket B das Ziel ist. Nehmen wir jetzt an, Sie fügen eine weitere Replikations-Konfiguration hinzu, bei der Bucket B die Quelle und Bucket C das Ziel ist. In diesem Fall werden Objekte in Bucket B, die Replikate von Objekten in Bucket A sind, nicht in Bucket C repliziert.
- Objekte im Quell-Bucket, die bereits auf ein anderes Ziel repliziert wurden. Wenn Sie beispielsweise den Ziel-Bucket in einer vorhandenen Replikations-Konfiguration ändern, repliziert S3 in Outposts diese Objekte nicht erneut.
- Objekte, die mit der serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) erstellt wurden.

- Aktualisierungen von Unterressourcen auf Bucket-Ebene.

  Wenn Sie beispielsweise die Lebenszyklus-Konfiguration ändern oder eine Benachrichtigungskonfiguration zu Ihrem Quell-Bucket hinzufügen, werden diese Änderungen nicht auf den Ziel-Bucket angewendet. Durch diese Funktion ist es möglich, für den Quell- und den Ziel-Bucket verschiedene Konfigurationen zu nutzen.
- Aktionen, die von der Lebenszyklus-Konfiguration durchgeführt werden.

  Wenn Sie beispielsweise eine Lebenszykluskonfiguration nur auf Ihrem Quell-Bucket aktivieren und Ablaufaktionen konfigurieren, erstellt S3 in Outposts Löschmarkierungen für abgelaufene Objekte im Quell-Bucket, repliziert diese Markierungen jedoch nicht in den Ziel-Bucket. Wenn Sie dieselbe Lebenszyklus-Konfiguration sowohl auf den Quell- als auch auf den Ziel-Bucket anwenden möchten, aktivieren Sie für beide Buckets dieselbe Lebenszyklus-Konfiguration. Weitere Informationen zur Lebenszyklus-Konfiguration finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.

Was wird von S3 Replication in Outposts nicht unterstützt?

Die folgenden Funktionen von S3 Replication werden von S3 in Outposts derzeit nicht unterstützt:

- Begrenzung der S3-Replikationszeit (S3 RTC) S3 RTC wird nicht unterstützt, da der Objektdatenverkehr in S3 Replication in Outposts über Ihr On-Premises-Netzwerk (das lokale Gateway) übertragen wird. Weitere Informationen zu lokalen Gateways finden Sie unter Arbeiten mit dem lokalen Gateway im Benutzerhandbuch zu AWS Outposts.
- S3 Replication für Batchvorgänge.

Einrichten der Replikation

> **Note**
>
> Objekte, die bereits vor dem Einrichten einer Replikationsregel in Ihrem Bucket vorhanden waren, werden nicht automatisch repliziert. Anders ausgedrückt: Amazon S3 in Outposts repliziert Objekte nicht rückwirkend. Um Objekte zu replizieren, die vor der Konfiguration Ihrer Replikation erstellt wurden, können Sie diese unter Verwendung der API-Operation `CopyObject` in denselben Bucket kopieren. Nach dem Kopieren werden die Objekte als „neue“ Objekte im Bucket angezeigt und es gilt die Replikationskonfiguration für diese Objekte. Weitere Informationen zum Kopieren eines Objekts finden Sie unter Kopieren eines

Objekts in einem Amazon S3 on Outposts-Bucket mit AWS SDK for Java und CopyObject in Amazon Simple Storage Service – API-Referenz.

Um die S3 Replication in Outposts zu aktivieren, fügen Sie Ihrem Quell-Outposts-Bucket eine Replikationsregel hinzu. Die Replikationsregel weist S3 in Outposts an, Objekte wie angegeben zu replizieren. In der Replikationsregel müssen Sie Folgendes angeben:

- Den Zugriffspunkt des Quell-Outposts-Buckets – Den Amazon-Ressourcennamen (ARN) des Zugriffspunkts oder den Zugriffspunktalias des Buckets, von dem aus S3 in Outposts die Objekte replizieren soll. Weitere Informationen zur Verwendung von Zugriffspunktaliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-in-Outposts-Buckets.
- Die Objekte, die Sie replizieren möchten – Sie können alle Objekte im Quell-Outposts-Bucket replizieren oder nur eine Teilmenge davon. Teilmengen identifizieren Sie, indem Sie ein Schlüsselnamenpräfix, mindestens ein Objekt-Tag oder beides in der Konfiguration angeben.

  Wenn Sie beispielsweise eine Replikationsregel konfigurieren, um nur Objekte mit dem Schlüsselnamenpräfix `Tax/` zu replizieren, repliziert S3 in Outposts Objekte mit Schlüsseln wie `Tax/doc1` oder `Tax/doc2`. Es repliziert aber keine Objekte mit dem Schlüssel `Legal/doc3`. Wenn Sie sowohl ein Präfix als auch mindestens ein Tag angeben, repliziert S3 in Outposts nur Objekte, die dieses Schlüsselpräfix und diese Tags aufweisen.
- Den Ziel-Outposts-Bucket – Den ARN oder Zugriffspunktalias des Buckets, in den S3 in Outposts die Objekte replizieren soll.

Sie können die Replikationsregel über die REST-API, AWS-SDKs, die AWS Command Line Interface (AWS CLI) oder die Amazon-S3-Konsole konfigurieren.

S3 in Outposts stellt auch API-Vorgänge zur Unterstützung der Einrichtung von Replikationsregeln bereit. Weitere Informationen finden Sie in den folgenden Themen in der Amazon Simple Storage Service – API-Referenz.

- PutBucketReplication
- GetBucketReplication
- DeleteBucketReplication

Themen

- Voraussetzungen für das Erstellen von Konfigurationsregeln
- Erstellen von Replikationsregeln in Outposts

Voraussetzungen für das Erstellen von Konfigurationsregeln

Themen

- Verbinden Ihrer Quell- und Ziel-Outpost-Subnetze
- Erstellen einer IAM-Rolle

Verbinden Ihrer Quell- und Ziel-Outpost-Subnetze

Damit Ihr Replikationsdatenverkehr über Ihr lokales Gateway von Ihrem Quell-Outpost zu Ihrem Ziel-Outpost geleitet wird, müssen Sie eine neue Route hinzufügen, um das Netzwerk einzurichten. Sie müssen die Classless Inter-Domain Routing (CIDR)-Netzwerkbereiche Ihrer Zugriffspunkte miteinander verbinden. Für jedes Zugriffspunktpaar müssen Sie diese Verbindung nur einmal einrichten.

Einige Schritte zum Einrichten der Verbindung unterscheiden sich je nach Zugriffstyp Ihrer Outposts-Endpunkte, die Ihren Zugriffspunkten zugeordnet sind. Der Zugriffstyp für Endpunkte ist entweder Privat (direktes Virtual Private Cloud [VPC]-Routing für AWS Outposts) oder Kundeneigene IP-Adresse (ein kundeneigener IP-Adresspool [CoIP-Pool] in Ihrem On-Premises-Netzwerk).

Schritt 1: Ermitteln des CIDR-Bereichs Ihres Quell-Outposts-Endpunkts

So ermitteln Sie den CIDR-Bereich Ihres Quellendpunkts, der Ihrem Quellzugriffspunkt zugeordnet ist

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie in der Liste Outposts-Buckets den gewünschten Quell-Bucket für die Replikation aus.
4. Wählen Sie die Registerkarte Outposts-Zugriffspunkte und anschließend den Outposts-Zugriffspunkt für den Quell-Bucket für Ihre Replikationsregel aus.
5. Wählen Sie den Outposts-Endpunkt aus.
6. Kopieren Sie die Subnetz-ID, die in Schritt 5 verwendet werden soll.
7. Die Methode, mit der Sie den CIDR-Bereich des Quell-Outposts-Endpunkts ermitteln, hängt vom Zugriffstyp Ihres Endpunkts ab.

Prüfen Sie im Abschnitt Outposts-Endpunkt – Übersicht den Zugriffstyp.

- Wenn der Zugriffstyp Privat lautet, kopieren Sie den Wert für Classless inter-domain routing (CIDR), der in Schritt 6 verwendet werden soll.
- Wenn der Zugriffstyp Kundeneigene IP-Adresse lautet, gehen Sie wie folgt vor:
  1. Kopieren Sie den Wert für Kundeneigener IPv4-Pool, um ihn später als ID des Adresspools zu verwenden.
  2. Öffnen Sie die AWS Outposts-Konsole unter https://console.aws.amazon.com/outposts/.
  3. Wählen Sie im Navigationsbereich Lokale Gateway-Routing-Tabellen aus.
  4. Wählen Sie den Wert für Lokale Gateway-Routing-Tabellen-ID Ihres Quell-Outposts aus.
  5. Wählen Sie im Detailbereich die Registerkarte CoIP-Pools aus. Fügen Sie den Wert Ihrer CoIP-Pool-ID, den Sie zuvor kopiert haben, in das Suchfeld ein.
  6. Kopieren Sie für den übereinstimmenden CoIP-Pool den entsprechenden CIDRs-Wert Ihres Quell-Outposts-Endpunkts, um ihn in Schritt 6 zu verwenden.

Schritt 2: Ermitteln der Subnetz-ID und des CIDR-Bereichs Ihres Ziel-Outposts-Endpunkts

Um die Subnetz-ID und den CIDR-Bereich Ihres Zielendpunkts zu ermitteln, der Ihrem Zielzugriffspunkt zugeordnet ist, führen Sie dieselben Unterschritte in Schritt 1 aus und ändern Sie dabei Ihren Quell-Outposts-Endpunkt in Ihren Ziel-Outposts-Endpunkt. Kopieren Sie den Subnetz-ID-Wert Ihres Ziel-Outposts-Endpunkts, um ihn in Schritt 6 zu verwenden. Kopieren Sie den CIDR-Wert Ihres Ziel-Outposts-Endpunkts, um ihn in Schritt 5 zu verwenden.

Schritt 3: Ermitteln der lokalen Gateway-ID Ihres Quell-Outposts

So ermitteln Sie die lokale Gateway-ID Ihres Quell-Outposts

1. Öffnen Sie die AWS Outposts-Konsole unter https://console.aws.amazon.com/outposts/.
2. Wählen Sie im linken Navigationsbereich Lokale Gateways aus.
3. Suchen Sie auf der Seite Lokale Gateways nach der Outpost-ID Ihres Quell-Outposts, den Sie für die Replikation verwenden möchten.
4. Kopieren Sie den Wert der lokalen Gateway-ID Ihres Quell-Outposts, um ihn in Schritt 5 zu verwenden.

Weitere Informationen zu lokalen Gateways finden Sie unter Lokales Gateway im AWS Outposts-Benutzerhandbuch.

Schritt 4: Ermitteln der lokalen Gateway-ID Ihres Ziel-Outposts

Um die lokale Gateway-ID Ihres Ziel-Outposts zu ermitteln, führen Sie dieselben Unterschritte in Schritt 3 aus, wobei Sie allerdings nach der Outpost-ID für Ihren Ziel-Outpost suchen. Kopieren Sie den Wert der lokalen Gateway-ID Ihres Ziel-Outposts, um ihn in Schritt 6 zu verwenden.

Schritt 5: Einrichten der Verbindung von Ihrem Quell-Outpost-Subnetz zu Ihrem Ziel-Outpost-Subnetz

So richten Sie eine Verbindung von Ihrem Quell-Outpost-Subnetz zu Ihrem Ziel-Outpost-Subnetz ein

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die VPC-Konsole unter https://console.aws.amazon.com/vpc/.
2. Wählen Sie im linken Navigationsbereich die Option Subnets aus.
3. Geben Sie im Suchfeld die Subnetz-ID für Ihren Quell-Outposts-Endpunkt ein, den Sie in Schritt 1 ermittelt haben. Wählen Sie das Subnetz mit der übereinstimmenden Subnetz-ID aus.
4. Wählen Sie für das übereinstimmende Subnetzelement den Wert für Routing-Tabelle dieses Subnetzes aus.
5. Wählen Sie auf der Seite mit ausgewählter Routing-Tabelle Aktionen und dann Routen bearbeiten aus.
6. Wählen Sie auf der Seite Routen bearbeiten die Option Route hinzufügen aus.
7. Geben Sie unter Ziel den CIDR-Bereich Ihres Ziel-Outposts-Endpunkts ein, den Sie in Schritt 2 ermittelt haben.
8. Wählen Sie unter Ziel Outpost lokales Gateway aus und geben Sie die lokale Gateway-ID Ihres Quell-Outposts ein, die Sie in Schritt 3 ermittelt haben.
9. Wählen Sie Änderungen speichern aus.
10. Vergewissern Sie sich, dass der Status für die Route Aktiv lautet.

Schritt 6: Einrichten der Verbindung von Ihrem Ziel-Outpost-Subnetz zu Ihrem Quell-Outpost-Subnetz

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die VPC-Konsole unter https://console.aws.amazon.com/vpc/.
2. Wählen Sie im linken Navigationsbereich die Option Subnets aus.

3. Geben Sie im Suchfeld die Subnetz-ID für Ihren Ziel-Outposts-Endpunkt ein, den Sie in Schritt 2 ermittelt haben. Wählen Sie das Subnetz mit der übereinstimmenden Subnetz-ID aus.
4. Wählen Sie für das übereinstimmende Subnetzelement den Wert für Routing-Tabelle dieses Subnetzes aus.
5. Wählen Sie auf der Seite mit ausgewählter Routing-Tabelle Aktionen und dann Routen bearbeiten aus.
6. Wählen Sie auf der Seite Routen bearbeiten die Option Route hinzufügen aus.
7. Geben Sie unter Ziel den CIDR-Bereich Ihres Ziel-Outposts-Endpunkts ein, den Sie in Schritt 1 ermittelt haben.
8. Wählen Sie unter Ziel Outpost lokales Gateway aus und geben Sie die lokale Gateway-ID Ihres Ziel-Outposts ein, die Sie in Schritt 4 ermittelt haben.
9. Wählen Sie Änderungen speichern aus.
10. Vergewissern Sie sich, dass der Status für die Route Aktiv lautet.

Nachdem Sie die CIDR-Netzwerkbereiche Ihrer Quell- und Zielzugriffspunkte verbunden haben, müssen Sie eine AWS Identity and Access Management (IAM)-Rolle erstellen.

Erstellen einer IAM-Rolle

Standardmäßig sind alle S3–in-Outputs-Ressourcen – Buckets, Objekte und zugehörige Unterressourcen – privat, sodass nur der Ressourcenbesitzer auf die Ressource zugreifen kann. S3 in Outputs benötigt Berechtigungen zum Lesen und Replizieren von Objekten aus dem Quell-Outposts-Bucket. Sie erteilen diese Berechtigungen, indem Sie eine IAM-Servicerolle erstellen und die Rolle in der Replikationskonfiguration festlegen.

In diesem Abschnitt werden die Vertrauensrichtlinie und die mindestens erforderliche Berechtigungsrichtlinie erläutert. Diese Beispielanleitungen enthalten Schritt-für-Schritt-Anweisungen zum Erstellen einer IAM-Rolle. Weitere Informationen finden Sie unter Erstellen von Replikationsregeln in Outposts. Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

- Im folgenden Beispiel ist eine Vertrauensrichtlinie zu sehen, bei der Sie S3 in Outposts als Service-Prinzipal identifizieren, der die Rolle übernehmen kann.

```
{
   "Version":"2012-10-17",
   "Statement":[
```

```
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

- Im folgenden Beispiel wird eine Zugriffsrichtlinie gezeigt, bei der Sie der Rolle die Berechtigungen erteilen, Replikationsaufgaben in Ihrem Namen durchzuführen. Wenn S3 in Outposts die Rolle annimmt, verfügt es über die Berechtigungen, die Sie in dieser Richtlinie angeben. Wenn Sie diese Richtlinie verwenden möchten, ersetzen Sie *`user input placeholders`* durch eigene Informationen. Stellen Sie sicher, dass Sie dabei die Outpost-IDs Ihrer Quell- und Ziel-Outposts sowie die Bucket-Namen und Zugriffspunktnamen Ihrer Quell- und Ziel-Outposts-Buckets verwenden.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:GetObjectVersionForReplication",
            "s3-outposts:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
bucket/SOURCE-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
accesspoint/SOURCE-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:ReplicateObject",
            "s3-outposts:ReplicateDelete"
         ],
         "Resource":[
```

```
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/
bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/
accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      }
   ]
}
```

Die Zugriffsrichtlinie erteilt Berechtigungen für folgenden Aktionen:

- `s3-outposts:GetObjectVersionForReplication` – Die Berechtigung für diese Aktion wird für alle Objekte erteilt, damit S3 in Outposts eine bestimmte Objektversion abrufen kann, die jedem Objekt zugeordnet ist.

- `s3-outposts:GetObjectVersionTagging` – Die Berechtigung für diese Aktion für Objekte im *`SOURCE-OUTPOSTS-BUCKET`*-Bucket (Quell-Bucket) gestattet es S3 in Outposts, Objekt-Tags für die Replikation zu lesen. Weitere Informationen finden Sie unter Hinzufügen von Tags für S3-on-Outposts-Buckets. Wenn S3 in Outposts nicht über diese Berechtigung verfügt, repliziert es die Objekte, nicht jedoch die Objekt-Tags.
- `s3-outposts:ReplicateObject` und `s3-outposts:ReplicateDelete` – Die Berechtigungen für diese Aktionen für alle Objekte im *`DESTINATION-OUTPOSTS-BUCKET`*-Bucket (Ziel-Bucket) erlauben es S3 in Outposts, Objekte oder Löschmarkierungen in den Ziel-Outposts-Bucket zu replizieren. Informationen zu Löschmarkierungen finden Sie unter Auswirkungen von LöschVorgänge auf die Replikation.

> **Note**
>
> - Die Berechtigung für die `s3-outposts:ReplicateObject`-Aktion im *`DESTINATION-OUTPOSTS-BUCKET`*-Bucket (Ziel-Bucket) erlaubt auch die Replikation von Objekt-Tags. Daher müssen Sie für die `s3-outposts:ReplicateTags`-Aktion keine explizite Berechtigung erteilen.
> - Für die kontoübergreifende Replikation muss der Besitzer des Ziel-Outposts-Buckets seine Bucket-Richtlinie aktualisieren, um die Berechtigung für die `s3-outposts:ReplicateObject`-Aktion in dem *`DESTINATION-OUTPOSTS-BUCKET`* zu erteilen. Die `s3-outposts:ReplicateObject`-Aktion ermöglicht es S3 in Outposts, Objekte und Objekt-Tags in den Ziel-Outposts-Bucket zu replizieren.

Eine Liste der Aktionen von S3 in Outposts finden Sie unter Aktionen, die von Amazon S3 in Outposts definiert werden.

> **Important**
>
> Das AWS-Konto, das die IAM-Rolle besitzt, muss über Berechtigungen für die Aktionen verfügen, die der IAM-Rolle erteilt werden.
> Angenommen, der Quell-Outposts-Bucket enthält Objekte, die im Besitz eines anderen AWS-Kontos sind. Der Eigentümer der Objekte muss dem AWS-Konto, das die IAM-Rolle besitzt, die erforderlichen Berechtigungen explizit über die Bucket-Richtlinie und die Zugriffspunktrichtlinie erteilen. Andernfalls kann S3 in Outposts nicht auf die Objekte zugreifen und die Replikation der Objekte schlägt fehl.
> Die hier beschriebenen Berechtigungen gehören zur Mindest-Replikationskonfiguration. Wenn Sie optionale Replikationskonfigurationen hinzufügen möchten, müssen Sie S3 in Outposts zusätzliche Berechtigungen erteilen.

Erteilen von Berechtigungen, wenn sich die Quell- und Ziel-Outposts-Buckets im Besitz verschiedener AWS-Konten befinden

Wenn sich die Quell- und Ziel-Outposts-Buckets nicht im Besitz desselben Kontos befinden, muss der Eigentümer des Ziel-Outposts-Buckets die Bucket- und Zugriffspunkt-Richtlinien für den Ziel-Bucket aktualisieren. Diese Richtlinien müssen dem Besitzer des Quell-Outposts-Buckets und der IAM-Servicerolle Berechtigungen zum Ausführen von Replikationsaktionen gewähren, wie in den folgenden Richtlinienbeispielen dargestellt. Andernfalls schlägt die Replikation fehl. In diesen Richtlinienbeispielen ist *`DESTINATION-OUTPOSTS-BUCKET`* der Ziel-Bucket. Wenn Sie diese Richtlinienbeispiele verwenden möchten, ersetzen Sie die *`user input placeholders`* durch Ihre Informationen.

Wenn Sie die IAM-Servicerolle manuell erstellen, legen Sie den Rollenpfad als `role/service-role/` fest. wie in den folgenden Richtlinienbeispielen dargestellt. Weitere Informationen finden Sie unter IAM ARNs im IAM-Benutzerhandbuch.

```
{
   "Version":"2012-10-17",
   "Id":"PolicyForDestinationBucket",
   "Statement":[
      {
```

```
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
            "s3-outposts:ReplicateDelete",
            "s3-outposts:ReplicateObject"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:DestinationBucket-account-
ID:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*"
         ]
      }

   ]
}
```

```
{
"Version":"2012-10-17",
   "Id":"PolicyForDestinationAccessPoint",
   "Statement":[
      {
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
            "s3-outposts:ReplicateDelete",
            "s3-outposts:ReplicateObject"
         ],
         "Resource" :[
            "arn:aws:s3-outposts:region:DestinationBucket-account-
ID:outpost/DESTINATION-OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/
object/*"
         ]
      }
   ]
}
```

> **Note**
>
> Wenn Objekte im Quell-Outposts-Bucket mit einem Tag versehen sind, beachten Sie Folgendes:
> Wenn der Eigentümer des Quell-Outposts-Buckets S3 in Outposts die Berechtigung für die Aktionen `s3-outposts:GetObjectVersionTagging` und `s3-outposts:ReplicateTags` zum Replizieren von Objekt-Tags (über die IAM-Rolle) erteilt, repliziert Amazon S3 die Tags zusammen mit den Objekten. Weitere Information zur IAM-Rolle finden Sie unter Erstellen einer IAM-Rolle.

Erstellen von Replikationsregeln in Outposts

Die S3-Replikation in Outposts ist eine automatische, asynchrone und Bucket-übergreifende Replikation von Objekten in demselben oder verschiedenen AWS Outposts. Bei der Replikation werden neu erstellte Objekte und Objektaktualisierungen aus einem Quell-Outposts-Bucket in einen oder mehrere Ziel-Outposts-Bucket(s) kopiert. Weitere Informationen finden Sie unter Replikation von Objekten für S3 in Outposts.

> **Note**
>
> Objekte, die bereits vor dem Einrichten von Replikationsregeln in Ihrem Quell-Outposts-Bucket vorhanden waren, werden nicht repliziert. Anders ausgedrückt: S3 in Outposts repliziert Objekte nicht rückwirkend. Um Objekte zu replizieren, die vor der Konfiguration Ihrer Replikation erstellt wurden, können Sie diese unter Verwendung der API-Operation `CopyObject` in denselben Bucket kopieren. Nach dem Kopieren werden die Objekte als „neue“ Objekte im Bucket angezeigt und es gilt die Replikationskonfiguration für diese Objekte. Weitere Informationen zum Kopieren eines Objekts finden Sie unter Kopieren eines Objekts in einem Amazon S3 on Outposts-Bucket mit AWS SDK for Java und CopyObject in Amazon Simple Storage Service – API-Referenz.

Wenn Sie die Replikation konfigurieren, fügen Sie dem Quell-Outposts-Bucket Replikationsregeln hinzu. Replikationsregeln definieren, welche Quell-Outposts-Bucket-Objekte repliziert werden sollen und in welchem Ziel-Outposts-Bucket/welchen Ziel-Outposts-Buckets die replizierten Objekte gespeichert werden sollen. Sie können eine Regel erstellen, um alle Objekte in einem Bucket oder eine Untermenge von Objekten mit einem spezifischen Schlüsselnamenpräfixen, einem oder

mehreren Objekt-Markierungen oder beidem zu replizieren. Ein Ziel-Outposts-Bucket kann sich in demselben Outpost wie der Quell-Outposts-Bucket oder in einem anderen Outpost befinden.

Für die Replikationsregeln von S3 in Outposts müssen Sie sowohl den Amazon-Ressourcennamen (ARN) des Zugriffspunkts des Quell-Outposts-Buckets als auch den ARN des Zugriffspunkts des Ziel-Outposts-Buckets anstelle der Namen des Quell-Outposts-Buckets und des Ziel-Outposts-Buckets angeben.

Wenn Sie angeben, dass eine Objektversions-ID gelöscht werden soll, löscht S3 in Outposts diese Objektversion im Quell-Outposts-Bucket. Die Löschung wird jedoch nicht in den Ziel-Outposts-Bucket repliziert. Anders ausgedrückt: Dieselbe Objektversion wird nicht aus dem Ziel-Outposts-Bucket gelöscht. Dieses Verhalten schützt Daten vor böswilligen Löschungen.

Wenn Sie einem Outposts-Bucket eine Replikationsregel hinzufügen, ist diese standardmäßig aktiviert, sodass sie ausgeführt wird, sobald Sie sie speichern.

In diesem Beispiel richten Sie eine Replikation für Quell- und Ziel-Outposts-Buckets ein, die sich in unterschiedlichen Outposts befinden und demselben AWS-Konto gehören. Beispiele für die Verwendung der Amazon-S3-Konsole, der AWS Command Line Interface (AWS CLI) und der AWS SDK for Java und AWS SDK for .NET. Informationen zu den kontoübergreifenden Berechtigungen für die S3-Replikation in Outposts finden Sie unter Erteilen von Berechtigungen, wenn sich die Quell- und Ziel-Outposts-Buckets im Besitz verschiedener AWS-Konten befinden.

Die Voraussetzungen für die Einrichtung der Replikationsregeln von S3 in Outposts finden Sie unter Voraussetzungen für das Erstellen von Konfigurationsregeln.

Verwenden der S3-Konsole

Führen Sie die folgenden Schritte aus, um eine Replikationsregel zu konfigurieren, wenn sich der Amazon-S3-in-Outposts-Ziel-Bucket in einem anderen Outpost als der Quell-Outposts-Bucket befindet.

Wenn sich der Ziel-Outposts-Bucket in einem anderen Konto als der Quell-Outposts-Bucket befindet, müssen Sie dem Ziel-Outposts-Bucket eine Bucket-Richtlinie hinzufügen, um dem Eigentümer des Quell-Outposts-Bucket-Kontos die Berechtigung zum Replizieren von Objekten in den Ziel-Outposts-Bucket zu erteilen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener befinden AWS-Konten.

So erstellen Sie eine Replikationsrolle

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Outposts-Buckets den Namen des Buckets aus, den Sie als Quell-Bucket verwenden möchten.
3. Wählen Sie die Registerkarte Verwaltung aus, scrollen Sie nach unten zum Abschnitt Replikationsregeln und wählen Sie dann Replikationsregel erstellen aus.
4. Geben Sie in Name der Replikationsregel einen Namen für Ihre Regel ein, um sie später besser identifizieren zu können. Der Name ist erforderlich und muss innerhalb des Buckets eindeutig sein.
5. In Status ist standardmäßig Aktiviert ausgewählt. Eine aktivierte Regel wird ausgeführt, sobald Sie speichern. Wenn Sie die Regel später aktivieren möchten, wählen Sie Deaktiviert aus.
6. Unter Priorität bestimmt der Prioritätswert der Regel, welche Regel im Falle einer Überschneidung von Regeln angewendet wird. Wenn Objekte in den Geltungsbereich von mehr als einer Replikationsregel fallen, verwendet S3 in Outposts diese Prioritätswerte, um Konflikte zu vermeiden. Standardmäßig werden neue Regeln mit der höchsten Priorität zur Replikationskonfiguration hinzugefügt. Je höher die Zahl, desto höher die Priorität.

   Um die Priorität für die Regel zu ändern, wählen Sie nach dem Speichern der Regel zunächst den Namen der Regel aus der Liste der Replikationsregeln, dann Aktionen und schließlich Priorität bearbeiten aus.
7. Unter Quell-Bucket stehen Ihnen folgende Optionen zum Festlegen der Replikationsquelle zur Verfügung:

   - Um den gesamten Bucket zu replizieren, wählen Sie Auf alle Objekte im Bucket anwenden aus.
   - Um die Präfix- oder Tag-Filterung auf die Replikationsquelle anzuwenden, wählen Sie Geltungsbereich dieser Regel durch Verwendung von einem oder mehreren Filtern beschränken aus. Sie können ein Präfix und Markierungen kombinieren.
     - Um alle Objekte mit demselben Präfix zu replizieren, geben Sie unter Präfix ein Präfix in das Feld ein. Bei Verwendung des Filters Präfix ist die Replikation auf alle Objekte beschränkt, deren Namen mit derselben Zeichenfolge beginnen (z. B. `pictures`).

       Wenn Sie ein Präfix eingeben, bei dem es sich um den Namen eines Ordners handelt, müssen Sie einen / (Schrägstrich) als letztes Zeichen eingeben (z. B. `pictures/`).

- Um alle Objekte mit einem oder mehreren gleichen Objekt-Tags zu replizieren, wählen Sie Tag hinzufügen aus und geben Sie das Schlüssel-Wert-Paar in die Felder ein. Wiederholen Sie den Vorgang, um ein weiteres Tag hinzuzufügen. Weitere Informationen über Objekt-Markierungen finden Sie unter Hinzufügen von Tags für S3-on-Outposts-Buckets.

8. Um für die Replikation auf Ihren S3-in-Outposts-Quell-Bucket zuzugreifen, wählen Sie unter Quellzugriffspunktname einen Zugriffspunkt aus, der an den Quell-Bucket angehängt ist.
9. Wählen Sie unter Ziel den Zugriffspunkt-ARN des Ziel-Outposts-Buckets aus, in den S3 in Outposts Objekte replizieren soll. Der Ziel-Outposts-Bucket kann sich in demselben AWS-Konto wie der Quell-Outposts-Bucket oder in einem anderen befinden.

   Wenn sich der Ziel-Bucket in einem anderen Konto als der Quell-Outposts-Bucket befindet, müssen Sie dem Ziel-Outposts-Bucket eine Bucket-Richtlinie hinzufügen, um dem Eigentümer des Quell-Outposts-Bucket-Kontos die Berechtigung zum Replizieren von Objekten in den Ziel-Outposts-Bucket zu erteilen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, wenn sich die Quell- und Ziel-Outposts-Buckets im Besitz verschiedener AWS-Konten befinden.

   > **Note**
   >
   > Wenn die Versionsverwaltung für den Ziel-Outposts-Bucket nicht aktiviert ist, erhalten Sie eine Warnmeldung, die die Schaltfläche Versionierung aktivieren enthält. Wählen Sie diese Schaltfläche, um das Versioning für den Bucket zu aktivieren.

10. Richten Sie eine AWS Identity and Access Management (IAM)-Servicerolle ein, die S3 in Outposts annehmen kann, um Objekte in Ihrem Namen zu replizieren.

    Führen Sie zum Einrichten einer IAM-Rolle unter IAM-Rolle einen der folgenden Schritte aus:

    - Damit S3 in Outposts eine neue IAM-Rolle für Ihre Replikationskonfiguration erstellt, wählen Sie Aus vorhandenen IAM-Rollen auswählen und dann Neue Rolle erstellen aus. Wenn Sie die Regel speichern, wird eine neue Richtlinie für die IAM-Rolle erstellt, die mit den von Ihnen ausgewählten Quell- und Ziel-Outposts-Buckets übereinstimmt. Wir empfehlen Ihnen, die Option Neue Rolle erstellen auszuwählen.
    - Sie haben auch die Möglichkeit, eine vorhandene IAM-Rolle zu verwenden. In diesem Fall müssen Sie eine Rolle auswählen, die S3 in Outposts die erforderlichen Berechtigungen für die Replikation gewährt. Wenn diese Rolle S3 in Outposts keine ausreichenden Berechtigungen gewährt, um Ihre Replikationsregel zu befolgen, schlägt die Replikation fehl.

Um eine vorhandene Rolle auszuwählen, wählen Sie Aus vorhandenen IAM-Rollen auswählen und dann im Dropdown-Menü die Rolle aus. Sie können auch die Option IAM-Rollen-ARN eingeben auswählen und dann den Amazon-Ressourcennamen (ARN) der IAM-Rolle eingeben.

> **Important**
>
> Wenn Sie eine Replikationsregel zu einem S3-in-Outposts-Bucket hinzufügen, benötigen Sie die Berechtigungen `iam:CreateRole` und `iam:PassRole`, um die IAM-Rolle erstellen und übergeben zu können, die S3 in Outposts Replikationsberechtigungen gewährt. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service übergeben kann im IAM-Benutzerhandbuch.

11. Alle Objekte in Outposts-Buckets sind standardmäßig verschlüsselt. Weitere Informationen zur Verschlüsselung in S3 in Outposts finden Sie unter Datenverschlüsselung in S3 on Outposts. Nur Objekte, die durch die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verschlüsselt wurden, können repliziert werden. Die Replikation von Objekten, die durch serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) oder durch serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) verschlüsselt wurden, wird nicht unterstützt.
12. Aktivieren Sie beim Festlegen der Konfiguration der Replikationsregeln nach Bedarf die folgenden zusätzlichen Optionen:

    - Wenn Sie S3-in-Outposts-Replikationsmetriken in Ihrer Replikationskonfiguration aktivieren möchten, wählen Sie Replikationsmetriken aus. Weitere Informationen finden Sie unter Überwachen des Fortschritts mit Replikationsmetriken.
    - Wenn Sie die Löschmarkierungs-Replikation in Ihrer Replikations-Konfiguration aktivieren möchten, wählen Sie Markierungsreplikation löschen aus. Weitere Informationen finden Sie unter Auswirkungen von LöschVorgänge auf die Replikation.
    - Wenn Sie an den Replikaten vorgenommene Metadatenänderungen zurück in die Quellobjekte replizieren möchten, wählen Sie Synchronisierung von Replikatänderungen aus. Weitere Informationen finden Sie unter Replikationsstatus, wenn die Synchronisierung von Amazon-S3-Replikatänderungen in Outposts aktiviert ist.
13. Wählen Sie Regel erstellen aus, um den Vorgang abzuschließen.

Nach dem Speichern der Regel können Sie diese bearbeiten, aktivieren, deaktivieren oder löschen. Wechseln Sie hierfür auf die Registerkarte Verwaltung für den Quell-Outposts-Bucket, scrollen Sie nach unten zum Abschnitt Replikationsregeln, wählen Sie Ihre Regel aus und wählen Sie dann Regel bearbeiten aus.

Verwendung von AWS CLI

Um die AWS CLI zum Einrichten der Replikation zu verwenden, wenn sich die Quell- und Ziel-Outposts-Buckets im Besitz desselben AWS-Kontos befinden, gehen Sie wie folgt vor:

- Erstellen Sie Quell- und Ziel-Outposts-Buckets.
- Aktivieren Sie die Versionsverwaltung für beide Buckets.
- Erstellen Sie eine IAM-Rolle, die S3 in Outposts die Berechtigung zur Replikation von Objekten erteilt.
- Fügen Sie die Replikationskonfiguration zum Quell-Outposts-Bucket hinzu.

Um die Einrichtung zu prüfen, testen Sie sie.

So richten Sie die Replikation ein, wenn sich die Quell- und Ziel-Outposts-Buckets im Besitz desselben AWS-Kontos befinden

1. Richten Sie das Anmeldeinformationsprofil für die AWS CLI ein. In diesem Beispiel verwenden wir den Profilnamen `acctA`. Informationen zum Einrichten der Anmeldeinformations-Profile finden Sie unter Named Profiles (Benannte Profile) im AWS Command Line Interface-Benutzerhandbuch.

   > **Important**
   >
   > Das Profil, das Sie für diese Übung verwenden, muss über die nötigen Berechtigungen verfügen. Beispielsweise legen Sie in der Replikationskonfiguration die IAM-Servicerolle fest, die S3 in Outposts annehmen kann. Dies können Sie nur tun, wenn das verwendete Profil über die Berechtigungen `iam:CreateRole` und `iam:PassRole` verfügt. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service übergeben kann im IAM-Benutzerhandbuch. Wenn Sie zur Erstellung eines benannten Profils die Anmeldeinformationen eines Administrators verwenden, verfügt das benannte Profil über die erforderliche Berechtigung, um alle Aufgaben durchzuführen.

2. Erstellen Sie einen Quell-Bucket und aktivieren Sie das Versioning für ihn. Mit dem folgenden Befehl `create-bucket` wird ein *SOURCE-OUTPOSTS-BUCKET*-Bucket in der Region USA Ost (Nord-Virginia) (`us-east-1`) erstellt. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

```
aws s3control create-bucket --bucket SOURCE-OUTPOSTS-BUCKET --outpost-id SOURCE-OUTPOST-ID --profile acctA --region us-east-1
```

Mit dem folgenden Befehl `put-bucket-versioning` wird die Versionsverwaltung auf dem *SOURCE-OUTPOSTS-BUCKET*-Bucket aktiviert. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/bucket/SOURCE-OUTPOSTS-BUCKET --versioning-configuration Status=Enabled --profile acctA
```

3. Erstellen Sie einen Ziel-Bucket und aktivieren Sie das Versioning für ihn. Mit dem folgenden Befehl `create-bucket` wird ein *DESTINATION-OUTPOSTS-BUCKET*-Bucket in der Region USA West (Oregon) (`us-west-2`) erstellt. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

> **Note**
>
> Um die Replikationskonfiguration einzurichten, wenn sich die Quell- und Ziel-Outposts-Bucket in demselben AWS-Konto befinden, verwenden Sie dasselbe benannte Profil. Dieses Beispiel verwendet `acctA`. Zum Testen der Replikationskonfiguration, wenn sich die Buckets im Besitz unterschiedlicher AWS-Konten befinden, legen Sie verschiedene Profile für jeden Bucket fest.

```
aws s3control create-bucket --bucket DESTINATION-OUTPOSTS-BUCKET --create-bucket-configuration LocationConstraint=us-west-2 --outpost-id DESTINATION-OUTPOST-ID --profile acctA --region us-west-2
```

Mit dem folgenden Befehl `put-bucket-versioning` wird die Versionsverwaltung auf dem *DESTINATION-OUTPOSTS-BUCKET*-Bucket aktiviert. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET --versioning-configuration Status=Enabled --profile acctA
```

4. Erstellen Sie eine IAM-Servicerolle. Zu einem späteren Zeitpunkt der Replikationskonfiguration fügen Sie diese Servicerolle dem *SOURCE-OUTPOSTS-BUCKET*-Bucket hinzu. S3 in Outposts übernimmt diese Rolle, um Objekte in Ihrem Namen zu replizieren. Sie erstellen eine IAM-Rolle in zwei Schritten:

   a. Erstellen Sie eine IAM-Rolle.

      i. Kopieren Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-on-outposts-role-trust-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie gewährt S3 in Outposts Service-Prinzipal-Berechtigungen, um die Servicerolle anzunehmen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

      ii. Führen Sie den folgenden -Befehl aus, um die Rolle zu erstellen. Ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws iam create-role --role-name replicationRole --assume-role-policy-document file://s3-on-outposts-role-trust-policy.json --profile acctA
```

   b. Fügen Sie eine Berechtigungsrichtlinie zur Servicerolle hinzu.

      i. Kopieren Sie die folgende Berechtigungsrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-on-outposts-role-permissions-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie erteilt Berechtigungen für

verschiedene S3-in-Outposts-Bucket- und -Objektaktionen. Wenn Sie diese Richtlinie verwenden möchten, ersetzen Sie *user input placeholders* durch eigene Informationen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:GetObjectVersionForReplication",
            "s3-outposts:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-
OUTPOST-ID/bucket/SOURCE-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-
OUTPOST-ID/accesspoint/SOURCE-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:ReplicateObject",
            "s3-outposts:ReplicateDelete"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-
OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-
OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      }
   ]
}
```

ii. Führen Sie den folgenden Befehl aus, um eine Richtlinie zu erstellen und sie der Rolle anzufügen. Ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
aws iam put-role-policy --role-name replicationRole --policy-
document file://s3-on-outposts-role-permissions-policy.json --policy-
name replicationRolePolicy --profile acctA
```

5. Fügen Sie eine Replikationskonfiguration zum *SOURCE-OUTPOSTS-BUCKET*-Bucket hinzu.

   a. Zwar erfordert die S3-in-Outposts-API eine Replikationskonfiguration im XML-Format, die AWS CLI verlangt jedoch die Angabe der Replikationskonfiguration im JSON-Format. Speichern Sie den folgenden JSON-Code in einer Datei mit dem Namen `replication.json` im lokalen Verzeichnis auf Ihrem Computer. Wenn Sie diese Konfiguration verwenden möchten, ersetzen Sie *user input placeholders* durch Ihre Informationen.

```
{
  "Role": "IAM-role-ARN",
  "Rules": [
    {
      "Status": "Enabled",
      "Priority": 1,
      "DeleteMarkerReplication": { "Status": "Disabled" },
      "Filter" : { "Prefix": "Tax"},
      "Destination": {
        "Bucket":
        "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-
ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT"
      }
    }
  ]
}
```

   b. Führen Sie den folgenden Befehl `put-bucket-replication` aus, um die Replikationskonfiguration zu Ihrem Quell-Outposts-Bucket hinzuzufügen. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

```
aws s3control put-bucket-replication --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-
ID/bucket/SOURCE-OUTPOSTS-BUCKET --replication-configuration file://
replication.json --profile acctA
```

   c. Um die Replikationskonfiguration abzurufen, verwenden Sie den Befehl `get-bucket-replication`. Zur Verwendung dieses Befehls ersetzen Sie *user input placeholders* durch eigene Informationen.

```
aws s3control get-bucket-replication --account-id 123456789012 --bucket
 arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
bucket/SOURCE-OUTPOSTS-BUCKET --profile acctA
```

6. Testen Sie das Setup in der Amazon-S3-Konsole:

   a. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.

   b. Erstellen Sie im *SOURCE-OUTPOSTS-BUCKET*-Bucket einen Ordner mit dem Namen `Tax`.

   c. Fügen Sie Beispielobjekte zum `Tax`-Ordner im *SOURCE-OUTPOSTS-BUCKET*-Bucket hinzu.

   d. Überprüfen Sie im *DESTINATION-OUTPOSTS-BUCKET*-Bucket Folgendes:

      - S3 in Outposts hat die Objekte repliziert.

        > **Note**
        >
        > Die von S3 in Outposts für die Replikation eines Objekts benötigte Zeit hängt von der Größe des Objekts ab. Weitere Informationen zum Anzeigen des Replikationsstatus finden Sie unter Abrufen von Replikationsstatusinformationen.

      - Auf der Registerkarte Eigenschaften des Objekts ist Replikationsstatus auf Replikat gesetzt (sodass dies als Replikatobjekt identifiziert wird).

Verwalten Ihrer Replikation

In diesem Abschnitt werden zusätzliche Optionen für die Replikationskonfiguration beschrieben, die in S3 in Outposts verfügbar sind, und es wird erörtert, wie Sie den Replikationsstatus ermitteln und Replikationsprobleme beheben können. Weitere Informationen zum Erstellen einer grundlegenden Replikationskonfiguration finden Sie unter Einrichten der Replikation.

Themen

- Überwachen des Fortschritts mit Replikationsmetriken
- Abrufen von Replikationsstatusinformationen
- Fehlerbehebung bei einer Replikation
- Verwenden von EventBridge für S3 Replication in Outposts

Überwachen des Fortschritts mit Replikationsmetriken

S3 Replication in Outposts bietet detaillierte Metriken für die Replikationsregeln in Ihrer Replikationskonfiguration. Mithilfe der Replikationsmetriken können Sie den Fortschritt der Replikation in 5-Minuten-Intervallen überwachen. Verfolgen Sie dazu die Bytes der ausstehenden Replikation, die Replikationslatenz und die Operationen mit ausstehender Replikation. Zur Unterstützung bei der Behebung von Konfigurationsproblemen können Sie Amazon EventBridge auch so einrichten, dass Nachrichten über Replikationsfehler erhalten werden.

Wenn Replikationsmetriken aktiviert sind, veröffentlicht S3 Replication in Outposts die folgenden Metriken in Amazon CloudWatch:

- Bytes der ausstehenden Replikation – Die Gesamtzahl der Bytes von Objekten, deren Replikation für eine bestimmte Replikationsregel aussteht.
- Replikationslatenz – Die maximale Anzahl von Sekunden, um die der Replikations-Ziel-Bucket für eine bestimmte Replikationsregel hinter dem Quell-Bucket zurückliegt.
- Operationen mit ausstehender Replikation – Die Anzahl der Operationen, deren Replikation für eine bestimmte Replikationsregel aussteht. Zu den Operationen gehören Objekte, Löschmarkierungen und Tags.

> **Note**
>
> Die Metriken von S3 Replication in Outposts werden zu demselben Preis abgerechnet wie benutzerdefinierte CloudWatch-Metriken. Weitere Informationen hierzu finden Sie unter Amazon CloudWatch – Preise.

Abrufen von Replikationsstatusinformationen

Der Replikationsstatus hilft Ihnen, den aktuellen Status eines Objekts zu bestimmen, das gerade von Amazon S3 in Outposts repliziert wird. Der Replikationsstatus eines Quellobjekts gibt entweder `PENDING`, `COMPLETED` oder `FAILED` zurück. Der Replikationsstatus eines Replikats gibt `REPLICA` zurück.

Übersicht über den Replikationsstatus

In einem Replikationsszenario haben Sie einen Quell-Bucket, auf dem Sie die Replikation konfigurieren, und einen Ziel-Bucket, in den S3 in Outposts Objekte repliziert. Wenn Sie ein Objekt

(unter Verwendung von `GetObject`) oder Objektmetadaten (unter Verwendung von `HeadObject`) von diesen Buckets anfordern, gibt S3 in Outposts den Header `x-amz-replication-status` wie folgt in der Antwort zurück:

- Wenn Sie ein Objekt aus dem Quell-Bucket anfordern, gibt S3 in Outposts den Header `x-amz-replication-status` zurück, wenn das Objekt in der Anforderung für die Replikation geeignet ist.

  Nehmen wir beispielsweise an, dass Sie in Ihrer Replikationskonfiguration das Objektpräfix `TaxDocs` angeben, um S3 in Outposts anzuweisen, nur Objekte mit dem Schlüsselnamenpräfix `TaxDocs` zu replizieren. Alle Objekte mit diesem Schlüsselnamenpräfix, die Sie hochladen, z. B. `TaxDocs/document1.pdf`, werden repliziert. Für Objektanforderungen mit diesem Schlüsselnamenpräfix gibt S3 in Outposts den Header `x-amz-replication-status` mit einem der folgenden Werte für den Replikationsstatus des Objekts zurück: `PENDING`, `COMPLETED` oder `FAILED`.

  > **Note**
  >
  > Wenn nach dem Hochladen eines Objekts die Objektreplikation fehlschlägt, können Sie die fehlgeschlagene Replikation nicht erneut durchzuführen versuchen. Sie müssen das Objekt erneut hochladen. Bei Problemen wie fehlenden Replikationsrollen-Berechtigungen oder fehlenden Bucket-Berechtigungen gehen Objekte in den Status `FAILED` über. Bei temporären Fehlern, z. B. wenn ein Bucket oder Ihr Outpost nicht verfügbar ist, geht der Replikationsstatus nicht in `FAILED` über, sondern verbleibt bei `PENDING`. Wenn die Ressource wieder online ist, setzt S3 in Outposts die Replikation dieser Objekte fort.

- Wenn Sie ein Objekt aus einem Ziel-Bucket anfordern und es sich bei dem Objekt in Ihrer Anforderung um ein Replikat handelt, das von S3 in Outposts erstellt wurde, gibt S3 in Outposts den Header `x-amz-replication-status` mit dem Wert `REPLICA` zurück.

> **Note**
>
> Bevor Sie ein Objekt aus einem Quell-Bucket löschen, bei dem die Replikation aktiviert ist, sollten Sie den Replikationsstatus des Objekts überprüfen, um sicherzustellen, dass das Objekt repliziert wurde.

Replikationsstatus, wenn die Synchronisierung von Amazon-S3-Replikatänderungen in Outposts aktiviert ist

Wenn in Ihren Replikationsregeln die Synchronisierung von S3-in-Outposts-Replikatänderungen aktiviert ist, können Replikate einen anderen Status als `REPLICA` melden. Wenn Änderungen an Metadaten gerade repliziert werden, gibt der Header `x-amz-replication-status` für das Replikat `PENDING` zurück. Wenn bei der Synchronisierung der Replikatänderungen die Replikation von Metadaten fehlschlägt, gibt der Header für das Replikat `FAILED` zurück. Wenn Metadaten korrekt repliziert werden, gibt der Header für das Replikat den Wert `REPLICA` zurück.

Fehlerbehebung bei einer Replikation

Wenn Objektreplikate nicht im Amazon-S3-in-Outposts-Ziel-Bucket angezeigt werden, nachdem Sie die Replikation konfiguriert haben, können Sie mit diesen Tipps zur Fehlerbehebung Probleme identifizieren und beheben.

- Wie lange Amazon S3 in Outposts für die Replikation eines Objekts benötigt, hängt von verschiedenen Faktoren ab, unter anderem von der Distanz zwischen den Quell- und Ziel-Outposts und der Größe des Objekts.

  Sie können den Replikationsstatus des Quellobjekts überprüfen. Wenn der Replikationsstatus des Objekts `PENDING` lautet, hat S3 in Outposts die Replikation noch nicht abgeschlossen. Wenn der Replikationsstatus des Objekts `FAILED` lautet, überprüfen Sie die Replikationskonfiguration des Quell-Buckets.
- Überprüfen Sie in der Replikations-Konfiguration des Quell-Buckets Folgendes:
  - ob der Amazon-Ressourcenname (ARN) des Zugriffspunkts des Ziel-Buckets korrekt ist.
  - ob das Schlüsselnamenpräfix korrekt ist. Wenn Sie beispielsweise die Konfiguration so einrichten, dass nur Objekte mit dem Präfix `Tax` repliziert werden, werden nur Objekte mit Schlüsselnamen wie beispielsweise `Tax/document1` oder `Tax/document2` repliziert. Ein Objekt mit dem Schlüsselnamen `document3` wird nicht repliziert.
  - Der Status laute `Enabled`.
- Stellen Sie sicher, dass die Versionsverwaltung bei keinem der beiden Buckets ausgesetzt wurde. Sowohl für die Quell- als auch für die Ziel-Buckets muss die Versionsverwaltung aktiviert sein.
- Wenn sich der Ziel-Bucket im Besitz eines anderen AWS-Konto befindet, stellen Sie sicher, dass der Bucket-Eigentümer eine Bucket-Richtlinie für den Ziel-Bucket eingerichtet hat, die dem Eigentümer des Quell-Buckets die Replikation von Objekten gestattet. Ein Beispiel finden Sie

unter Erteilen von Berechtigungen, wenn sich die Quell- und Ziel-Outposts-Buckets im Besitz verschiedener AWS-Konten befinden.

- Wenn ein Objektreplikat nicht im Ziel-Bucket angezeigt wird, könnte Folgendes die Replikation verhindern:
  - S3 in Outposts repliziert keine Objekte in einem Quell-Bucket, bei dem es sich um ein Replikat handelt, das mit einer anderen Replikationskonfiguration erstellt wurde. Wenn Sie beispielsweise ein Replikationskonfiguration von Bucket A zu Bucket B zu Bucket C festlegen, repliziert S3 in Outposts keine Objektreplikate in Bucket B zu Bucket C.

    Wenn Sie Objekte in Bucket A zu Bucket B und Bucket C replizieren möchten, legen Sie mehrere Bucket-Ziele in unterschiedlichen Replikationsregeln für Ihre Quell-Bucket-Replikationskonfiguration fest. Erstellen Sie beispielsweise zwei Replikationsregeln für Quell-Bucket A, wobei eine Regel für die Replikation in Ziel-Bucket B und die andere Regel für die Replikation in Ziel-Bucket C gilt.
  - Ein Quell-Bucket-Eigentümer kann anderen AWS-Konten Berechtigungen für das Hochladen von Objekten erteilen. Standardmäßig besitzt der Quell-Bucket-Eigentümer keine Berechtigungen für die Objekte, die von anderen Konten erstellt wurden. Die Replikations-Konfiguration repliziert nur die Objekte, für die der Quell-Bucket-Eigentümer über Zugriffsberechtigungen verfügt. Um Replikationsprobleme zu vermeiden, kann der Quell-Bucket-Eigentümer anderen AWS-Konten Berechtigungen zum bedingten Erstellen von Objekten erteilen. Dabei sind explizite Zugriffsberechtigungen für diese Objekte erforderlich. Eine Beispielrichtlinie finden Sie unter Erteilung von kontoübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt.
- Angenommen, Sie fügen einer Replikationskonfiguration eine Regel hinzu, um eine Teilmenge von Objekten mit einem spezifischen Tag zu replizieren. In diesem Fall müssen Sie den spezifischen Tag-Schlüssel und -Wert zum Zeitpunkt der Objekterstellung zuweisen, damit S3 in Outposts das Objekt replizieren kann. Wenn Sie zuerst ein Objekt erstellen und dann dem vorhandenen Objekt das Tag hinzufügen, repliziert S3 in Outposts das Objekt nicht.
- Die Replikation schlägt fehl, wenn die Bucket-Richtlinie den Zugriff auf die Replikationsrolle für eine der folgenden Aktionen verweigert:

  Quell-Bucket

```
"s3-outposts:GetObjectVersionForReplication",
"s3-outposts:GetObjectVersionTagging"
```

Ziel-Buckets:

```
"s3-outposts:ReplicateObject",
"s3-outposts:ReplicateDelete",
"s3-outposts:ReplicateTags"
```

- Amazon EventBridge kann Sie darüber informieren, wenn Objekte nicht in ihre Ziel-Outposts repliziert werden. Weitere Informationen finden Sie unter Verwenden von EventBridge für S3 Replication in Outposts.

Verwenden von EventBridge für S3 Replication in Outposts

Amazon S3 in Outposts ist in Amazon EventBridge integriert und verwendet den Namespace `s3-outposts`. EventBridge ist ein Serverless-Event-Bus-Service, mit dem Sie Ihre Anwendungen mit Daten aus verschiedenen Quellen verbinden können. Weitere Informationen finden Sie unter Was ist Amazon EventBridge? im Amazon-EventBridge-Benutzerhandbuch.

Zur Unterstützung bei der Behebung von Problemen mit der Replikationskonfiguration können Sie Amazon EventBridge so einrichten, dass Nachrichten über Replikationsfehlerereignisse erhalten werden. EventBridge kann Sie darüber informieren, wenn Objekte nicht in ihre Ziel-Outposts repliziert werden. Weitere Informationen zum aktuellen Status eines zu replizierenden Objekts finden Sie unter Übersicht über den Replikationsstatus.

S3 in Outposts kann Ereignisse an EventBridge senden, wenn bestimmte Ereignisse in Ihrem Outposts-Bucket stattfinden. Anders als bei anderen Zielen müssen Sie nicht auswählen, welche Ereignistypen Sie liefern möchten. Sie können EventBridge-Regeln auch verwenden, um Ereignisse an zusätzliche Ziele weiterzuleiten. Nachdem EventBridge aktiviert wurde, sendet S3 in Outposts alle folgenden Ereignisse an EventBridge.

Ereignistyp	Beschreibung	Namespace
`Operation FailedReplication`	Die Replikation eines Objekts innerhalb einer Replikationsregel ist fehlgeschlagen. Weitere Informationen darüber, warum S3 Replication in Outposts fehlgeschlagen ist, finden Sie unter Verwenden von EventBrid	`s3-outposts`

Ereignistyp	Beschreibung	Namespace
	ge zur Anzeige der Ursachen von Fehlern bei S3 Replication in Outposts.	

Verwenden von EventBridge zur Anzeige der Ursachen von Fehlern bei S3 Replication in Outposts

In der folgenden Tabelle sind Gründe für das Fehlschlagen von S3 Replication in Outposts aufgeführt. Sie können eine EventBridge-Regel so konfigurieren, dass die Fehlerursache über Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS), AWS Lambda oder Amazon CloudWatch Logs veröffentlicht und angezeigt wird. Weitere Informationen zu den erforderlichen Berechtigungen zur Verwendung dieser Ressourcen für EventBridge finden Sie unter Verwenden von ressourcenbasierten Richtlinien für EventBridge.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`AssumeRoleNotPermitted`	S3 in Outposts kann die in der Replikationskonfiguration angegebene AWS Identity and Access Management (IAM)-Rolle nicht übernehmen.
`DstBucketNotFound`	S3 in Outposts kann den in der Replikationskonfiguration angegebenen Ziel-Bucket nicht finden.
`DstBucketUnversioned`	Die Versionsverwaltung ist im Outposts-Ziel-Bucket nicht aktiviert. Um Objekte mit S3 Replication in Outposts replizieren zu können, müssen Sie die Versionsverwaltung im Ziel-Bucket aktivieren.
`DstDelObjNotPermitted`	S3 in Outposts kann Löschvorgänge nicht in den Ziel-Bucket replizieren. Möglicherweise fehlt die `s3-outposts:ReplicateDelete` -Berechtigung für den Ziel-Bucket.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`DstMultipartCompleteNotPermitted`	S3 in Outposts kann einen mehrteiligen Upload von Objekten in den Ziel-Bucket nicht abschließen. Möglicherweise fehlt die `s3-outposts:ReplicateObject` -Berechtigung für den Ziel-Bucket.
`DstMultipartInitNotPermitted`	S3 in Outposts kann einen mehrteiligen Upload von Objekten in den Ziel-Bucket nicht initiieren. Möglicherweise fehlt die `s3-outposts:ReplicateObject` -Berechtigung für den Ziel-Bucket.
`DstMultipartPartUploadNotPermitted`	S3 in Outposts kann keine mehrteiligen Objekte in den Ziel-Bucket hochladen . Möglicherweise fehlt die `s3-outposts:ReplicateObject` -Berechtigung für den Ziel-Bucket.
`DstOutOfCapacity`	S3 in Outposts kann nicht in den Ziel-Outpost replizieren, da die S3-Speicherkapazität des Outposts aufgebraucht ist.
`DstPutObjNotPermitted`	S3 in Outposts kann keine Objekte in den Ziel-Bucket replizieren. Möglicherweise fehlt die `s3-outposts:ReplicateObject` -Berechtigung für den Ziel-Bucket.
`DstPutTaggingNotPermitted`	S3 in Outposts kann keine Objekt-Tags in den Ziel-Bucket replizieren. Möglicherweise fehlt die `s3-outposts:ReplicateObject` -Berechtigung für den Ziel-Bucket.
`DstVersionNotFound`	S3 in Outposts kann die Objektversion, die benötigt wird, um die Metadaten dieser Objektversion zu replizieren, nicht im Ziel-Bucket finden.

Gründe für das Fehlschlagen der Replikation	Beschreibung
`SrcBucketReplicationConfigMissing`	S3 in Outposts kann keine Replikationskonfiguration für den Zugriffspunkt finden, der dem Quell-Outposts-Bucket zugeordnet ist.
`SrcGetObjNotPermitted`	S3 in Outposts kann nicht auf das Objekt im Quell-Bucket für die Replikation zugreifen. Möglicherweise fehlt die `s3-outposts:GetObjectVersionForReplication` -Berechtigung für den Quell-Bucket.
`SrcGetTaggingNotPermitted`	S3 in Outposts kann nicht auf Objekt-Tag-Informationen vom Quell-Bucket zugreifen. Möglicherweise fehlt die `s3-outposts:GetObjectVersionTagging` -Berechtigung für den Quell-Bucket.
`SrcHeadObjectNotPermitted`	S3 in Outposts kann keine Objektmetadaten aus dem Quell-Bucket abrufen. Möglicherweise fehlt die `s3-outposts:GetObjectVersionForReplication` -Berechtigung für den Quell-Bucket.
`SrcObjectNotEligible`	Das Objekt kann nicht repliziert werden. Das Objekt oder seine Objekt-Tags stimmt/stimmen nicht mit der Replikationskonfiguration überein.

Weitere Informationen zur Behebung von Replikationsfehlern finden Sie in folgenden Themen:

- Erstellen einer IAM-Rolle
- Fehlerbehebung bei einer Replikation

Überwachen von EventBridge mit CloudWatch

Für die Überwachung lässt sich Amazon EventBridge mit Amazon CloudWatch integrieren. EventBridge sendet automatisch jede Minute Metriken an CloudWatch. Zu diesen Metriken gehören die Anzahl der Ereignisse, die von einer Regel abgeglichen wurden, sowie die Anzahl der Aufrufe eines Ziels durch eine Regel. Wenn eine Regel in EventBridge ausgeführt wird, werden alle Ziele aufgerufen, die mit der Regel verknüpft sind. Sie können das Verhalten von EventBridge wie folgt mit CloudWatch überwachen.

- Sie können die verfügbaren EventBridge-Metriken für Ihre EventBridge-Regeln über das CloudWatch-Dashboard überwachen. Anschließend können Sie CloudWatch-Funktionen wie beispielsweise CloudWatch-Alarme verwenden, um Alarme für bestimmte Metriken einzustellen. Wenn diese Metriken die benutzerdefinierten Schwellenwerte erreichen, die Sie in den Alarmen angegeben haben, erhalten Sie Benachrichtigungen und können entsprechende Maßnahmen ergreifen.
- Sie können Amazon CloudWatch Logs als Ziel Ihrer EventBridge-Regel festlegen. EventBridge erstellt dann Protokollstreams und CloudWatch Logs speichert den Text der Ereignisse als Protokolleinträge. Weitere Informationen finden Sie unter EventBridge und CloudWatch Logs.

Weitere Informationen zum Debuggen von EventBridge-Ereignisübermittlungs- und -archivierungsereignissen finden Sie unter den folgenden Themen:

- Richtlinie zur Wiederholung von Ereignissen und Verwendung von Warteschlangen für unzustellbare Nachrichten
- Archivieren von EventBridge-Ereignissen

Freigabe von S3 in Outposts mithilfe von AWS RAM

Amazon S3 in Outposts unterstützt die gemeinsame Nutzung von S3-Kapazitäten für mehrere Konten innerhalb einer Organisation mithilfe von AWS Resource Access Manager (AWS RAM). Mit der Freigabe von S3 in Outposts können Sie anderen erlauben, Buckets, Endpunkte und Zugriffspunkte in Ihrem Outpost zu erstellen und zu verwalten.

In diesem Thema wird veranschaulicht, wie Sie AWS RAM verwenden, um S3 in Outposts und verwandte Ressourcen für ein anderes AWS-Konto in Ihrer AWS-Organisation freizugeben.

Voraussetzungen

- Für das Outpost-Eigentümerkonto ist eine Organisation in AWS Organizations konfiguriert. Weitere Informationen finden Sie unter Erstellen einer Organisation im Benutzerhandbuch für AWS Organizations.
- Die Organisation umfasst das AWS-Konto, mit dem Sie Ihre Kapazität von S3 in Outposts teilen möchten. Weitere Informationen finden Sie unter Senden von Einladungen an AWS-Konten im Benutzerhandbuch für AWS Organizations.
- Wählen Sie eine der folgenden Optionen, die Sie freigeben möchten. Die zweite Ressource (entweder Subnets (Subnetze) oder Outposts) muss ausgewählt sein, damit auch Endpunkte zugänglich sind. Endpunkte sind eine Netzwerkanforderung, um auf Daten zuzugreifen, die in S3 in Outposts gespeichert sind.

Option 1	Option 2
S3 in Outposts Erlaubt es dem Benutzer, Buckets auf Ihren Outposts und Zugriffspunkten zu erstellen und diesen Buckets Objekte hinzuzufügen.	S3 in Outposts Erlaubt es dem Benutzer, Buckets auf Ihren Outposts und Zugriffspunkten zu erstellen und diesen Buckets Objekte hinzuzufügen.
Subnets Erlaubt es dem Benutzer, Ihre Virtual Private Cloud (VPC) und die Endpunkte zu verwenden, die mit Ihrem Subnetz verknüpft sind.	Outposts Erlaubt dem Benutzer das Anzeigen von S3-Kapazitätstabellen und der AWS Outposts-Konsolen-Startseite. Erlaubt es Benutzern außerdem, Subnetze auf freigegebenen Outposts zu erstellen und Endpunkte zu erstellen.

Verfahren

1. Melden Sie sich bei der AWS Management Console mit dem AWS-Konto an, dem der Outpost gehört, und öffnen Sie dann die AWS RAM-Konsole unter https://console.aws.amazon.com/ram.

2. Stellen Sie sicher, dass Sie die Freigabe mit AWS Organizations in AWS RAM aktiviert haben. Weitere Informationen finden Sie unter Freigabe für Ressourcen in AWS Organizations aktivieren im AWS RAM-Benutzerhandbuch.
3. Verwenden Sie entweder Option 1 oder Option 2 in den Voraussetzungen, um eine Ressourcenfreigabe zu erstellen. Wenn Sie mehrere S3-in-Outposts-Ressourcen haben, wählen Sie die Amazon-Ressourcennamen (ARNs) der Ressourcen aus, die Sie freigeben möchten. Wenn Sie Endpunkte aktivieren möchten, teilen Sie entweder Ihr Subnetz oder Ihren Outpost.

   Weitere Informationen zum Erstellen einer Ressourcenfreigabe finden Sie unter Erstellen einer Ressourcenfreigabe im AWS RAM-Benutzerhandbuch.
4. Das AWS-Konto, mit dem Sie Ihre Ressourcen geteilt haben, sollte jetzt in der Lage sein, S3 in Outposts zu verwenden. Abhängig von der Option, die Sie in den Voraussetzungen gewählt haben, geben Sie dem Kontobenutzer die folgenden Informationen an:

Option 1	Option 2
Die Outpost-ID	Die Outpost-ID
Die VPC-ID	
Die Subnetz-ID	
Die Sicherheitsgruppen-ID	

> **Note**
>
> Der Benutzer kann mithilfe der AWS RAM-Konsole, der AWS Command Line Interface (AWS CLI), der AWS-SDKs oder der REST-API bestätigen, dass die Ressourcen für ihn freigegeben wurden. Der Benutzer kann seine vorhandenen Ressourcenfreigaben anzeigen, indem er den CLI-Befehl get-resource-shares verwendet.

Verwendungsbeispiele

Nachdem Sie Ihre S3 in Outposts-Ressourcen mit einem anderen Konto geteilt haben, kann dieses Konto Buckets und Objekte in Ihrem Outpost verwalten. Wenn Sie die Ressource Subnets (Subnetze) freigegeben haben, kann dieses Konto den von Ihnen erstellten Endpunkt verwenden. Die folgenden

Beispiele veranschaulichen, wie ein Benutzer die AWS CLI verwendet, um mit Ihrem Outpost zu interagieren, nachdem Sie diese Ressourcen freigegeben haben.

Example : Erstellen eines Buckets

Im folgenden Beispiel wird ein Bucket namens *DOC-EXAMPLE-BUCKET1* im Outpost *op-01ac5d28a6a232904* erstellt. Bevor Sie diesen Befehl verwenden, ersetzen Sie jeden *user input placeholder* mit den entsprechenden Werten für Ihren Anwendungsfall.

```
aws s3control create-bucket --bucket DOC-EXAMPLE-BUCKET1 --outpost-
id op-01ac5d28a6a232904
```

Weitere Informationen über diesen Befehl finden Sie unter create-bucket in der AWS CLI-Referenz.

Example : Erstellen eines Zugriffspunkts

Im folgenden Beispiel wird ein Zugriffspunkt in einem Outpost erstellt, wobei die Beispielparameter in der folgenden Tabelle verwendet werden. Bevor Sie diesen Befehl verwenden, ersetzen Sie diese *user input placeholder*-Werte und den AWS-Region-Code mit den entsprechenden Werten für Ihren Anwendungsfall.

Parameter	Wert
Konto-ID	*111122223333*
Name des Zugriffspunkts	*example-outpost-access-point*
Outpost-ID	*op-01ac5d28a6a232904*
Name des Outpost-Buckets	*DOC-EXAMPLE-BUCKET1*
VPC-ID	*vpc-1a2b3c4d5e6f7g8h9*

> **Note**
>
> Der Konto-ID-Parameter muss die AWS-Konto-ID des Bucket-Eigentümers sein, der der freigegebene Benutzer ist.

```
aws s3control create-access-point --account-id 111122223333 --name example-outpost-access-point \
--bucket arn:aws:s3-outposts:us-east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/DOC-EXAMPLE-BUCKET1 \
--vpc-configuration VpcId=vpc-1a2b3c4d5e6f7g8h9
```

Weitere Informationen über diesen Befehl finden Sie unter create-access-point in der AWS CLI-Referenz.

Example : Hochladen eines Objekts

Im folgenden Beispiel wird die Datei *my_image.jpg* vom lokalen Dateisystem des Benutzers zu einem Objekt namens *images/my_image.jpg* durch den Zugriffspunkt *example-outpost-access-point* unter dem Outpost *op-01ac5d28a6a232904*, im Besitz des AWS-Kontos *111122223333* hochgeladen. Bevor Sie diesen Befehl verwenden, ersetzen Sie diese *user input placeholder*-Werte und den AWS-Region-Code mit den entsprechenden Werten für Ihren Anwendungsfall.

```
aws s3api put-object --bucket arn:aws:s3-outposts:us-east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-point \
--body my_image.jpg --key images/my_image.jpg
```

Weitere Informationen über diesen Befehl finden Sie unter put-object in der AWS CLI-Referenz.

> **Note**
>
> Wenn dieser Vorgang zu einem Fehler Ressource nicht gefunden führt oder nicht reagiert, verfügt Ihre VPC möglicherweise nicht über einen freigegebenen Endpunkt.
> Verwenden Sie den AWS CLI-Befehl list-shared-endpoints, um zu überprüfen, ob es einen freigegebenen Endpunkt gibt. Wenn kein freigegebener Endpunkt vorhanden ist, erstellen Sie einen Endpunkt gemeinsam mit dem Outpost-Besitzer. Weitere Informationen finden Sie unter ListSharedEndpoints in der API-Referenz zu Amazon Simple Storage Service.

Example : Erstellen eines Endpunkts

Das folgende Beispiel erstellt einen Endpunkt für einen freigegebenen Outpost. Bevor Sie diesen Befehl verwenden, ersetzen Sie die *user input placeholder*-Werte für die Outpost-

ID, die Subnetz-ID und die Sicherheitsgruppen-ID durch die entsprechenden Werte für Ihren Anwendungsfall.

> **Note**
>
> Der Benutzer kann diesen Vorgang nur ausführen, wenn die Ressourcenfreigabe die Outposts-Ressource enthält.

```
aws s3outposts create-endpoint --outposts-id op-01ac5d28a6a232904 --subnet-id XXXXXX --
security-group-id XXXXXXX
```

Weitere Informationen über diesen Befehl finden Sie unter create-endpoint in der AWS CLI-Referenz.

Sonstige AWS-Services, die S3 on Outposts verwenden

Andere AWS-Services, die lokal auf Ihrem AWS Outposts ausgeführt werden, können Ihre Kapazität von Amazon S3 on Outposts ebenfalls verwenden. In Amazon CloudWatch zeigt der Namespace `S3Outposts` detaillierte Metriken für Buckets in S3 on Outposts an. Diese Metriken berücksichtigt jedoch keine Nutzung anderer AWS-Services. Informationen zum Verwalten Ihrer S3-on-Outposts-Kapazität, die von anderen AWS-Services verbraucht wird, finden Sie in der folgenden Tabelle.

AWS-Service	Beschreibung	Weitere Informationen
Amazon S3	Die gesamte direkte S3-on-Outposts-Nutzung verfügt über eine CloudWatch-Metrik zum übereinstimmenden Konto und Bucket.	Siehe Metriken
Amazon Elastic Block Store (Amazon EBS)	Für Amazon EBS on Outposts können Sie einen AWS-Outpost als Snapshot-Ziel auswählen und in Ihrem S3 on Outpost lokal speichern.	Weitere Informationen
Amazon Relational Database Service (Amazon RDS)	Sie können lokale Amazon-RDS-Backups verwenden, um Ihre RDS-Backups lokal in Ihrem Outpost zu speichern.	Weitere Informationen

Überwachen von S3 in Outposts

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (OUTPOSTS). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Weitere Informationen zum Überwachen Ihrer Speicherkapazität von Amazon S3 in Outposts finden Sie in den folgenden Themen.

Themen

- Verwalten der Kapazität von S3 in Outposts mit Amazon-CloudWatch-Metriken
- Empfangen von S3-in-Outposts-Ereignisbenachrichtigungen mit Amazon CloudWatch Events
- Überwachen von S3 in Outposts mit Protokollen in AWS CloudTrail

Verwalten der Kapazität von S3 in Outposts mit Amazon-CloudWatch-Metriken

Zur besseren Verwaltung der festgelegten S3-Kapazität auf Ihrem Outpost sollten Sie CloudWatch-Warnungen erstellen, die Sie warnen, wenn die Speichernutzung einen bestimmten Schwellenwert überschreitet. Weitere Informationen zu CloudWatch-Metriken für S3 on Outposts finden Sie unter CloudWatch-Metriken. Wenn nicht genügend Speicherplatz vorhanden ist, um ein Objekt in Ihrem Outpost zu speichern, gibt die API eine Ausnahme wegen unzureichender Kapazität (ICE) zurück. Wenn Sie Speicherplatz freigeben möchten, können Sie CloudWatch-Alarme erstellen, die eine explizite Datenlöschung auslösen, oder eine Lebenszyklusablaufrichtlinie verwenden, um Objekte ablaufen zu lassen. Wenn Sie Daten vor dem Löschen speichern möchten, können AWS DataSync verwenden, um Daten aus Ihrem Bucket von Amazon S3 on Outposts in einen S3-Bucket in einer AWS-Region zu kopieren. Weitere Informationen zur Verwendung von DataSync finden Sie unter Erste Schritte mit AWS DataSync im AWS DataSync-Benutzerhandbuch.

CloudWatch-Metriken

Der `S3Outposts` Namespace enthält die folgenden Metriken für Amazon S3 auf Outposts-Buckets. Sie können die Gesamtzahl der bereitgestellten S3 in Outposts-Bytes, die für Objekte insgesamt verfügbaren freien Bytes und die Gesamtgröße aller Objekte für einen bestimmten Bucket überwachen. Bucket- oder kontobezogene Metriken gibt es für die gesamte direkte S3-Nutzung. Die indirekte S3-Nutzung, wie das Speichern lokaler Snapshots von Amazon Elastic Block Store oder Backups von Amazon Relational Database Service auf einem Outpost, verbraucht S3-Kapazität, ist aber nicht in den Bucket- oder kontobezogenen Metriken enthalten. Weitere Informationen über lokale Amazon-EBS-Snapshots finden Sie unter Amazon EBS local snapshots on Outposts. Ihren Amazon-EBS-Kostenbericht finden Sie unter https://console.aws.amazon.com/billing/.

> **Note**
>
> S3 in Outposts unterstützt nur die folgenden Metriken und keine anderen Amazon-S3-Metriken.
> Da S3 on Outposts über ein festgelegtes Kapazitätslimit verfügt, können Sie CloudWatch-Alarme erstellen, die Sie benachrichtigen, wenn die Speichernutzung einen bestimmten Schwellenwert überschreitet.

Metrik	Beschreibung	Zeitraum	Einheiten	Typ
`Outpost` `talByte`	Die gesamte bereitgestellte Kapazität in Byte für einen Outpost	5 Minuten	Bytes	S3 on Outposts
`Outpost` `eeBytes`	Die Anzahl der freien Bytes, die auf Outposts zum Speichern von Kundendaten verfügbar sind.	5 Minuten	Bytes	S3 on Outposts
`BucketU` `dBytes`	Die Gesamtgröße aller Objekte für den angegebenen Bucket.	5 Minuten	Bytes	S3 on Outposts Nur direkte S3-Nutzung

Metrik	Beschreibung	Zeitraum	Einheiten	Typ
AccountUsedBytes	Die Gesamtgröße aller Objekte für das angegebene Outposts-Konto.	5 Minuten	Bytes	S3 on Outposts Nur direkte S3-Nutzung
BytesPendingReplication	Die Gesamtanzahl der Bytes von Objekten, deren Replikation für eine bestimmte Replikationsregel aussteht. Weitere Informationen zum Aktivieren von Replikationsmetriken finden Sie unter Erstellen von Replikationsregeln zwischen Outposts.	5 Minuten	Bytes	Optional. Für S3 Replication in Outposts.
OperationsPendingReplication	Die Gesamtanzahl der Operationen, deren Replikation für eine bestimmte Replikationsregel aussteht. Weitere Informationen zum Aktivieren von Replikationsmetriken finden Sie unter Erstellen von Replikationsregeln zwischen Outposts.	5 Minuten	Zählunger	Optional. Für S3 Replication in Outposts.
ReplicationLatency	Die aktuelle Verzögerung in Sekunden, um die der Replikationsziel-Bucket hinter dem Quell-Bucket für eine bestimmte Replikationsregel zurückliegt. Weitere Informationen zum Aktivieren von Replikationsmetriken finden Sie unter Erstellen von Replikationsregeln zwischen Outposts.	5 Minuten	Sekunden	Optional. Für S3 Replication in Outposts.

Empfangen von S3-in-Outposts-Ereignisbenachrichtigungen mit Amazon CloudWatch Events

Sie können CloudWatch Events verwenden, um eine Regel für beliebige API-Ereignisse in Amazon S3 in Outposts zu erstellen. Wenn Sie eine Regel erstellen, können Sie sich über alle unterstützten Ziele von CloudWatch benachrichtigen lassen, einschließlich Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS) und AWS Lambda. Weitere Informationen finden Sie in der Liste der AWS-Services, die Ziele für CloudWatch Events sein können, im Amazon-CloudWatch-Events-Benutzerhandbuch. Informationen zur Auswahl eines Zieldienstes für die Zusammenarbeit mit S3 on Outposts finden Sie unter Erstellen einer CloudWatch-Events-Regel, die bei einem AWS-API-Aufruf mit AWS CloudTrail ausgelöst wird im Amazon-CloudWatch-Events-Benutzerhandbuch.

> **Note**
>
> Bei S3-in-Outposts-Objektoperationen stimmen die von CloudTrail gesendeten Aufrufereignisse der AWS-API nur dann mit Ihren Regeln überein, wenn Sie Trails (optional mit Ereignis-Selektoren) für den Empfang dieser Ereignisse konfiguriert haben. Weitere Informationen finden Sie unter Arbeiten mit CloudTrail-Protokolldateien im AWS CloudTrail-Benutzerhandbuch.

Example

Es folgt eine Beispielregel für den `DeleteObject`-Vorgang. Zum Verwenden dieser Beispielregel ersetzen Sie *`DOC-EXAMPLE-BUCKET1`* durch den Namen Ihres S3-in-Outposts-Buckets.

```
{
  "source": [
    "aws.s3-outposts"
  ],
  "detail-type": [
    "AWS API call through CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "s3-outposts.amazonaws.com"
    ],
    "eventName": [
      "DeleteObject"
```

```
    ],
    "requestParameters": {
      "bucketName": [
        "DOC-EXAMPLE-BUCKET1"
      ]
    }
  }
}
```

Überwachen von S3 in Outposts mit Protokollen in AWS CloudTrail

Amazon S3 in Outposts ist in AWS CloudTrail integriert. Dieser Service stellt eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem AWS-Service-Service in S3 durchgeführten Aktionen bereit. Sie können mit AWS CloudTrail Informationen über Anforderungen auf Bucket- und Objektebene in S3 in Outposts abrufen, um Ihre S3-in-Outposts-Ereignisaktivitäten zu überprüfen und zu protokollieren. Um CloudTrail-Datenereignisse für alle Outposts-Buckets oder für eine Liste spezifischer Outposts-Buckets zu aktivieren, müssen Sie manuell einen Trail in CloudTrail erstellen. Weitere Informationen zu CloudTrail-Protokolldateieinträgen finden Sie unter S3-in-Outposts-Protokolldateieinträge.

> **Note**
>
> - Eine bewährte Vorgehensweise besteht darin, eine Lebenszyklusrichtlinie für das AWS CloudTrail-Datenereignis für Ihren Outposts-Bucket zu erstellen. Konfigurieren Sie die Lebenszyklusrichtlinie zum regelmäßigen Entfernen von Protokolldateien nach dem Zeitraum, der für die Überprüfung erforderlich ist. Dadurch wird die Menge der Daten reduziert, die Amazon Athena in einer Abfrage analysiert. Weitere Informationen finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.
> - Beispiele für die Abfrage von CloudTrail-Protokollen finden Sie im AWS Big-Data-Blogbeitrag Analyze Security, Compliance and Operational Activity Using AWS CloudTrail and Amazon Athena.

CloudTrail-Protokollierung für Objekte in einem S3-in-Outposts-Bucket aktivieren

Sie können die Amazon-S3-Konsole verwenden, um einen AWS CloudTrail-Trail zum Protokollieren von Datenereignissen für Objekte in einem Amazon-S3-in-Outposts-Bucket zu konfigurieren. CloudTrail unterstützt die Protokollierung von API-Anforderungen auf Objektebene in S3 in Outposts

wie beispielsweise `GetObject`, `DeleteObject` und `PutObject`. Diese Ereignisse werden als Datenereignisse bezeichnet.

Standardmäßig werden Datenereignisse nicht von den CloudTrail-Trails protokolliert. Sie können Trails jedoch so konfigurieren, dass sie Datenereignisse für von Ihnen festgelegte S3-in-Outposts-Buckets protokollieren oder dass sie Datenereignisse für alle S3-in-Outposts-Buckets in Ihrem AWS-Konto protokollieren. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail.

CloudTrail gibt keine Datenereignisse in die CloudTrail-Ereignishistorie ein. Darüber hinaus werden nicht alle API-Operationen auf Bucket-Ebene in S3 in Outposts im CloudTrail-Ereignisverlauf aufgeführt. Weitere Informationen zum Abfragen von CloudTrail-Protokollen finden Sie unter Verwendung von Amazon-CloudWatch-Logs-Filtermustern und Amazon Athena zum Abfragen von CloudTrail-Protokollen im AWS-Wissenszentrum.

Um einen Trail zum Protokollieren von Datenereignissen für einen S3-in-Outposts-Bucket zu konfigurieren, können Sie entweder die AWS CloudTrail-Konsole oder die Amazon-S3-Konsole verwenden. Wenn Sie einen Trail zum Protokollieren von Datenereignissen für alle S3-in-Outposts-Buckets in Ihrem AWS-Konto konfigurieren möchten, ist es einfacher, die CloudTrail-Konsole zu verwenden. Informationen zur Verwendung der CloudTrail-Konsole zum Konfigurieren eines Trails zur Protokollierung von S3-in-Outposts-Datenereignissen finden Sie unter Daten-Ereignisse im Benutzerhandbuch zu AWS CloudTrail.

> **Important**
>
> Für Datenereignisse werden zusätzliche Gebühren fällig. Weitere Informationen finden Sie unter AWS CloudTrail – Preise.

Das folgende Verfahren zeigt, wie mit der Amazon-S3-Konsole ein CloudTrail-Trail so konfiguriert wird, dass er Datenereignisse für einen S3-in-Outposts-Bucket protokolliert.

> **Note**
>
> Das AWS-Konto, das den Bucket erstellt, besitzt ihn und ist das einzige, das S3-in-Outposts-Datenereignisse konfigurieren kann, die an AWS CloudTrail gesendet werden.

Aktivieren der Protokollierung von CloudTrail-Datenereignissen für Objekte in einem S3-in-Outposts-Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im Navigationsbereich Outposts buckets (Outposts-Buckets) aus.
3. Wählen Sie den Namen des Outposts-Buckets aus, dessen Datenereignisse Sie mit CloudTrail protokollieren möchten.
4. Wählen Sie Properties (Eigenschaften).
5. Wählen Sie im Abschnitt AWS CloudTrail-Datenereignisse die Option In CloudTrail konfigurieren aus.

   Die AWS CloudTrail-Konsole wird geöffnet.

   Sie können einen neuen CloudTrail-Trail erstellen oder einen vorhandenen Trail wiederverwenden und S3-in-Outposts-Datenereignisse so konfigurieren, dass sie in Ihrem Trail protokolliert werden.
6. Wählen Sie auf der Seite Dashboard der CloudTrail-Konsole die Option Trail erstellen aus.
7. Geben Sie auf der Seite Schritt 1 Trail-Attribute auswählen einen Namen für den Trail ein, wählen Sie einen S3-Bucket als Speicherort für die Trail-Protokolle aus, geben Sie alle weiteren gewünschten Einstellungen an und wählen Sie dann Nächstes aus.
8. Wählen Sie auf der Seite Schritt 2 Protokollereignisse auswählen unter Ereignistyp die Option Datenereignisse aus.

   Wählen Sie als Datenereignistyp S3 Outposts aus. Wählen Sie Next (Weiter).

   > **Note**
   >
   > - Wenn Sie einen Trail erstellen und die Datenereignisprotokollierung für S3 in Outposts konfigurieren, müssen Sie den Datenereignistyp korrekt angeben.
   >   - Wenn Sie die CloudTrail-Konsole verwenden, wählen Sie S3 Outposts als Datenereignistyp aus. Informationen zum Erstellen von Trails in der CloudTrail-Konsole finden Sie unter Erstellen und Aktualisieren eines Trails mit der Konsole im AWS CloudTrail-Benutzerhandbuch. Informationen zum Konfigurieren der S3-in-Outposts-Datenereignisprotokollierung in der CloudTrail-Konsole finden Sie unter

Protokollieren von Datenereignissen für Amazon-S3-Objekte im Benutzerhandbuch zu AWS CloudTrail.

- Wenn Sie die AWS Command Line Interface (AWS CLI) oder die AWS-SDKs verwenden, legen Sie für das Feld `resources.type` `AWS::S3Outposts::Object` fest. Weitere Informationen zum Protokollieren von S3-in-Outposts-Datenereignissen mit der AWS CLI finden Sie unter Protokollieren von S3-in-Outposts-Ereignissen im Benutzerhandbuch zu AWS CloudTrail.

- Wenn Sie die CloudTrail-Konsole oder die Amazon-S3-Konsole verwenden, um einen Trail zur Protokollierung von Datenereignissen für einen S3-in-Outposts-Bucket zu konfigurieren, zeigt die Amazon-S3-Konsole an, dass die Protokollierung auf Objektebene für den Bucket aktiviert ist.

9. Überprüfen Sie auf der Seite Schritt 3 Überprüfen und erstellen die von Ihnen konfigurierten Trail-Attribute und Protokollereignisse. Wählen Sie dann Trail erstellen aus.

Deaktivieren der Protokollierung von CloudTrail-Datenereignissen für Objekte in einem S3-in-Outposts-Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die CloudTrail-Konsole unter https://console.aws.amazon.com/cloudtrail/.
2. Wählen Sie im linken Navigationsbereich Trails aus.
3. Wählen Sie den Namen des Trails aus, den Sie erstellt haben, um Ereignisse für den S3-in-Outposts-Bucket zu protokollieren.
4. Wählen Sie oben rechts auf der Detailseite des Trails Protokollierung beenden aus.
5. Wählen Sie im anschließend angezeigten Dialogfeld Protokollierung beenden aus.

Entwickeln mit Amazon S3 on Outposts

Mit Amazon S3 in Outposts können Sie S3-Buckets auf Ihren AWS-Outposts erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresidenz erfordern, auf einfache Weise On-Premise speichern und abrufen. S3 on Outposts bietet eine neue Speicherklasse namens S3 Outposts (`OUTPOSTS`). Sie nutzt die Amazon-S3-APIs und ist darauf ausgelegt, Daten über mehrere Geräte und Server hinweg in Ihrem AWS Outposts dauerhaft und redundant zu speichern. Sie kommunizieren mit Ihrem Outpost-Bucket über einen Zugriffspunkt und eine Endpunktverbindung über eine Virtual Private Cloud (VPC). Sie können bei

Outpost-Buckets dieselben APIs und Funktionen wie in Amazon-S3-Buckets verwenden, inklusive Zugriffsrichtlinien, Verschlüsselung und Markierungen. Sie können S3 on Outposts über die AWS Management Console, AWS Command Line Interface (AWS CLI), AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Was ist Amazon S3 on Outposts?.

Die folgenden Themen enthalten Informationen zur Entwicklung mit S3 on Outposts.

Themen

- API-Vorgänge in Amazon S3 on Outposts
- Konfigurieren des S3-Steuerungsclients für S3 on Outposts mit dem SDK for Java
- Senden von Anforderungen an S3 on Outposts über IPv6

API-Vorgänge in Amazon S3 on Outposts

In diesem Thema werden die API-Vorgänge für Amazon S3, Amazon S3 Control und Amazon S3 on Outposts aufgeführt, die Sie mit Amazon S3 on Outposts verwenden können.

Themen

- Amazon-S3-API-Vorgänge für die Objektverwaltung
- Amazon-S3-Control-API-Vorgänge zum Verwalten von Buckets
- S3-on-Outposts-API-Vorgänge zur Verwaltung von Outposts

Amazon-S3-API-Vorgänge für die Objektverwaltung

S3 on Outposts ist so konzipiert, dass es die gleichen Objekt-API-Vorgänge wie Amazon S3 verwendet. Sie müssen Zugriffspunkte verwenden, um auf ein Objekt in einem Outpost-Bucket zuzugreifen. Wenn Sie eine Objekt-API-Operation mit S3 on Outposts verwenden, geben Sie entweder den Amazon-Ressourcennamen (ARN) des Zugriffspunkts für Outposts oder den Zugriffspunkt-Alias an. Weitere Informationen zu Zugriffspunkt-Aliassen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.

Amazon S3 on Outposts unterstützt die folgenden Amazon-S3-API-Operationen:

- AbortMultipartUpload
- CompleteMultipartUpload
- CopyObject

- CreateMultipartUpload
- DeleteObject
- DeleteObjects
- DeleteObjectTagging
- GetObject
- GetObjectTagging
- HeadBucket
- HeadObject
- ListMultipartUploads
- ListObjects
- ListObjectsV2
- ListObjectVersions
- ListParts
- PutObject
- PutObjectTagging
- UploadPart
- UploadPartCopy

Amazon-S3-Control-API-Vorgänge zum Verwalten von Buckets

S3 on Outposts unterstützt die folgenden Amazon-S3-Control-API-Vorgänge für die Arbeit mit Buckets.

- CreateAccessPoint
- CreateBucket
- DeleteAccessPoint
- DeleteAccessPointPolicy
- DeleteBucket
- DeleteBucketLifecycleConfiguration
- DeleteBucketPolicy
- DeleteBucketReplication
- DeleteBucketTagging

- GetAccessPoint
- GetAccessPointPolicy
- GetBucket
- GetBucketLifecycleConfiguration
- GetBucketPolicy
- GetBucketReplication
- GetBucketTagging
- GetBucketVersioning
- ListAccessPoints
- ListRegionalBuckets
- PutAccessPointPolicy
- PutBucketLifecycleConfiguration
- PutBucketPolicy
- PutBucketReplication
- PutBucketTagging
- PutBucketVersioning

S3-on-Outposts-API-Vorgänge zur Verwaltung von Outposts

S3 on Outposts unterstützt die folgenden API-Vorgänge für Amazon S3 on Outposts zur Verwaltung von Endpunkten.

- CreateEndpoint
- DeleteEndpoint
- ListEndpoints
- ListOutpostsWithS3
- ListSharedEndpoints

Konfigurieren des S3-Steuerungsclients für S3 on Outposts mit dem SDK for Java

Im folgenden Beispiel wird der Amazon-S3-Steuerungs-Client für Amazon S3 on Outposts mithilfe von AWS SDK for Java konfiguriert. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

```
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;

public AWSS3Control createS3ControlClient() {

    String accessKey = AWSAccessKey;
    String secretKey = SecretAccessKey;
    BasicAWSCredentials awsCreds = new BasicAWSCredentials(accessKey, secretKey);

    return AWSS3ControlClient.builder().enableUseArnRegion()
            .withCredentials(new AWSStaticCredentialsProvider(awsCreds))
            .build();

}
```

Senden von Anforderungen an S3 on Outposts über IPv6

Dual-Stack-Endpunkte von Amazon S3 on Outposts und S3 on Outposts unterstützen Anfragen an S3-on-Outposts-Buckets entweder über das IPv6- oder IPv4-Protokoll. Mit IPv6-Unterstützung für S3 on Outposts können Sie über S3-on-Outposts-APIs über IPv6-Netzwerke auf Ihre Buckets und Steuerebenenressourcen zugreifen und diese betreiben.

> **Note**
>
> Objektaktionen von S3 on Outposts (wie `PutObject` oder `GetObject`) werden über IPv6-Netzwerke nicht unterstützt.

Für den Zugriff auf S3 on Outposts über IPv6-Netzwerke fallen keine zusätzlichen Gebühren an. Weitere Informationen zu S3 on Outposts finden Sie unter S3 on Outposts – Preise.

Themen

Erste Schritte mit IPv6

Um eine Anforderung über IPv6 an einen S3-on-Outposts-Bucket zu stellen, müssen Sie einen Dual-Stack-Endpunkt verwenden. Der nächste Abschnitt beschreibt Anfragen über IPv6 unter Verwendung von Dual-Stack-Endpunkten.

Die folgenden wichtigen Überlegungen sind zu beachten, bevor Sie versuchen, über IPv6 auf einen S3-on-Outposts-Bucket zuzugreifen:

- Der Client und das Netzwerk, die auf den Bucket zugreifen, müssen für IPv6 aktiviert sein.
- Für den IPv6-Zugriff werden Anforderungen im virtuellen Hosting- und im Pfad-Stil unterstützt. Weitere Informationen finden Sie unter Verwenden von S3-on-Outposts-Dual-Stack-Endpunkten.
- Wenn Sie die Quell-IP-Adressfilterung in Ihren AWS Identity and Access Management (IAM)-Benutzer- oder S3-on-Outposts-Bucket-Richtlinien verwenden, müssen Sie die Richtlinien so aktualisieren, dass sie IPv6-Adressbereiche enthalten.

  > **Note**
  >
  > Diese Anforderung gilt nur für S3-on-Outposts-Bucket-Operationen und Ressourcen der Steuerebene in allen IPv6-Netzwerken. Objektaktionen von Amazon S3 on Outposts werden in IPv6-Netzwerken nicht unterstützt.

- Bei Verwendung von IPv6 geben die Serverzugriff-Protokolldateien IP-Adressen in einem IPv6-Format aus. Sie müssen vorhandene Tools, Skripts und Software aktualisieren, die Sie zum Analysieren von S3-on-Outposts-Protokolldateien verwenden, damit sie die IPv6-formatierten Remote-IP-Adressen analysieren können. Die aktualisierten Tools, Skripts und Software analysieren dann die IPv6-formatierten Remote-IP-Adressen korrekt.

Verwenden von Dual-Stack-Endpunkten für Anfragen über ein IPv6-Netzwerk

Um Anforderungen mit S3-on-Outposts-API-Aufrufen über IPv6 zu stellen, können Sie Dual-Stack-Endpunkte über AWS CLI oder AWS SDK verwenden. Die Amazon S3-Steuerungs-API-Operationen und S3-on-Outposts-API-Operationen funktionieren auf die gleiche Weise, unabhängig davon, ob Sie über ein IPv6-Protokoll oder ein IPv4-Protokoll auf S3 on Outposts zugreifen. Beachten Sie jedoch, dass Objektaktionen von S3 on Outposts (z. B. `PutObject` oder `GetObject`) über IPv6-Netzwerke nicht unterstützt werden.

Wenn Sie die AWS Command Line Interface (AWS CLI) und AWS SDKs verwenden, können Sie einen Parameter oder ein Flag verwenden, um zu einem Dual-Stack-Endpunkt zu wechseln. Sie können den Dual-Stack-Endpunkt auch direkt als Überschreibung des S3-on-Outposts-Endpunkts in der Konfigurationsdatei angeben.

Sie können einen Dual-Stack-Endpunkt verwenden, um über IPv6 von einer der folgenden Optionen auf einen S3-on-Outposts-Bucket zuzugreifen:

- Die AWS CLI; siehe Verwenden von Dual-Stack-Endpunkten von der AWS CLI.
- Die AWS-SDKs finden Sie unter Verwenden von S3-on-Outposts-Dual-Stack-Endpunkten über die AWS -SDKs.

Verwenden von IPv6-Adressen in IAM-Richtlinien

Bevor Sie versuchen, mit einem IPv6-Protokoll auf einen S3-on-Outposts-Bucket zuzugreifen, stellen Sie sicher, dass IAM-Benutzer oder S3-on-Outposts-Bucket-Richtlinien, die für die IP-Adressfilterung verwendet werden, aktualisiert werden, um IPv6-Adressbereiche einzuschließen. Wenn IP-Adressenfilterrichtlinien nicht für die Verarbeitung von IPv6-Adressen aktualisiert werden, können Sie den Zugriff auf einen S3-on-Outposts-Bucket verlieren, während Sie versuchen, das IPv6-Protokoll zu verwenden.

IAM-Richtlinien, die IP-Adressen filtern, verwenden Bedingungsoperatoren für IP-Adressen . Die folgende Bucket-Richtlinie für S3 on Outposts identifiziert die 54.240.143.* IP-Bereich zulässiger IPv4-Adressen unter Verwendung von Bedingungsoperatoren für IP-Adressen. Alle IP-Adressen außerhalb dieses Bereichs erhalten keinen Zugriff auf den S3-on-Outposts-Bucket (`DOC-EXAMPLE-BUCKET`). Alle IPv6-Adressen liegen außerhalb des zulässigen Bereichs, deshalb verhindert diese Richtlinie, dass IPv6-Adressen auf zugreife `DOC-EXAMPLE-BUCKET`.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3outposts:*",
      "Resource": "arn:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/DOC-EXAMPLE-BUCKET/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
      }
    }
  ]
}
```

Sie können das -`Condition`Element der S3-on-Outposts-Bucket-Richtlinie ändern, um sowohl IPv4- (`54.240.143.0/24`) als auch IPv6-Adressbereiche (`2001:DB8:1234:5678::/64`) zuzulassen, wie im folgenden Beispiel gezeigt. Sie können denselben Typ `Condition`-Block verwenden, wie im Beispiel gezeigt, um Ihre IAM-Benutzer- und Bucket-Richtlinien zu aktualisieren.

```
       "Condition": {
         "IpAddress": {
            "aws:SourceIp": [
              "54.240.143.0/24",
               "2001:DB8:1234:5678::/64"
             ]
          }
        }
```

Bevor Sie IPv6 verwenden, müssen Sie alle relevanten IAM-Benutzer- und S3-Bucket-Richtlinien aktualisieren, die eine IP-Adressfilterung verwenden, um die IPv6-Adressbereiche zu berücksichtigen. Wir empfehlen Ihnen, Ihre IAM-Richtlinien mit den IPv6-Adressbereichen Ihres Unternehmens zu aktualisieren, ebenso wie mit Ihren vorhandenen IPv4-Adressbereichen. Ein Beispiel für eine Bucket-Richtlinie, die den Zugriff über IPv6 und IPv4 gestattet, finden Sie unter Beschränken des Zugriffs auf bestimmte IP-Adressen.

Sie können Ihre IAM-Benutzerrichtlinien mit der IAM-Konsole unter https://console.aws.amazon.com/iam/ überprüfen. Weitere Informationen zu IAM finden Sie im IAM-Benutzerhandbuch. Informationen zum Bearbeiten von S3-on-Outposts-Bucket-Richtlinien finden Sie unter Hinzufügen oder Bearbeiten einer Bucket-Richtlinie für einen Amazon-S3-on-Outposts-Bucket.

Testen der IP-Adresskompatibilität

Wenn Sie eine Linux- oder Unix-Instance oder macOS X-Plattform verwenden, können Sie Ihren Zugriff auf einen Dual-Stack-Endpunkt über IPv6 testen. Um beispielsweise die Verbindung zu Amazon S3-on-Outposts-Endpunkten über IPv6 zu testen, verwenden Sie den `dig` Befehl :

```
dig s3-outposts.us-west-2.api.aws AAAA +short
```

Wenn Ihr Dual-Stack-Endpunkt über ein IPv6-Netzwerk ordnungsgemäß eingerichtet ist, gibt der `dig` Befehl die verbundenen IPv6-Adressen zurück. Beispielsweise:

```
dig s3-outposts.us-west-2.api.aws AAAA +short

2600:1f14:2588:4800:b3a9:1460:159f:ebce

2600:1f14:2588:4802:6df6:c1fd:ef8a:fc76

2600:1f14:2588:4801:d802:8ccf:4e04:817
```

Verwenden von IPv6 mit AWS PrivateLink

S3 on Outposts unterstützt das IPv6-Protokoll für -AWS PrivateLinkServices und -Endpunkte. Mit AWS PrivateLink Unterstützung für das IPv6-Protokoll können Sie über IPv6-Netzwerke eine Verbindung zu Service-Endpunkten innerhalb Ihrer VPC herstellen, entweder von lokalen oder anderen privaten Verbindungen. Die IPv6-Unterstützung für AWS PrivateLink für S3 on Outposts ermöglicht Ihnen auch die Integration AWS PrivateLink mit Dual-Stack-Endpunkten. Schritte zur Aktivierung von IPv6 für AWS PrivateLinkfinden Sie unter Beschleunigen Sie Ihre IPv6-Einführung mit -AWS PrivateLinkServices und -Endpunkten.

> **Note**
>
> Informationen zum Aktualisieren des unterstützten IP-Adresstyps von IPv4 auf IPv6 finden Sie unter Ändern des unterstützten IP-Adresstyps im AWS PrivateLink -Benutzerhandbuch.

Verwenden von IPv6 mit AWS PrivateLink

Wenn Sie AWS PrivateLink mit IPv6 verwenden, müssen Sie einen IPv6- oder Dual-Stack-VPC-Schnittstellenendpunkt erstellen. Allgemeine Schritte zum Erstellen eines VPC-Endpunkts mithilfe der

AWS Management Consolefinden Sie unter Zugriff auf einen -AWSService über einen Schnittstellen-VPC-Endpunkt im AWS PrivateLink -Benutzerhandbuch.

AWS Management Console

Gehen Sie wie folgt vor, um einen Schnittstellen-VPC-Endpunkt zu erstellen, der eine Verbindung zu S3 on Outposts herstellt.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die VPC-Konsole unter https://console.aws.amazon.com/vpc/.
2. Wählen Sie im Navigationsbereich Endpunkte aus.
3. Wählen Sie Endpunkt erstellen.
4. Wählen Sie bei Service category (Servicekategorie) die Option AWS services (-Services) aus.
5. Wählen Sie für Servicename den Service S3 on Outposts aus (com.amazonaws.us-east-1.s3-outposts).
6. Wählen Sie für VPC die VPC aus, von der aus Sie auf S3 on Outposts zugreifen.
7. Wählen Sie für Subnetze ein Subnetz pro Availability Zone aus, von dem aus Sie auf S3 on Outposts zugreifen. Sie können nicht mehrere Subnetze aus derselben Availability Zone auswählen. Für jedes von Ihnen ausgewählte Subnetz wird eine neue Endpunktnetzwerkschnittstelle erstellt. Standardmäßig werden IP-Adressen aus den Subnetz-IP-Adressbereichen den Endpunktnetzwerkschnittstellen zugewiesen. Um eine IP-Adresse für eine Endpunkt-Netzwerkschnittstelle festzulegen, wählen Sie IP-Adressen festlegen und geben Sie eine IPv6-Adresse aus dem Subnetz-Adressbereich ein.
8. Wählen Sie als IP-Adresstyp Dualstack aus. Weisen Sie Ihren Endpunktnetzwerkschnittstellen sowohl IPv4- als auch IPv6-Adressen zu. Diese Option wird nur unterstützt, wenn alle ausgewählten Subnetze sowohl IPv4- als auch IPv6-Adressbereiche aufweisen.
9. Wählen Sie für Sicherheitsgruppen die Sicherheitsgruppen aus, die den Endpunktnetzwerkschnittstellen für den VPC-Endpunkt zugeordnet werden sollen. Standardmäßig ist die Standardsicherheitsgruppe der VPC zugeordnet.
10. Wählen Sie für Richtlinie Vollzugriff, um alle Operationen aller Prinzipale auf allen Ressourcen über den VPC-Endpunkt zuzulassen. Wählen Sie andernfalls Benutzerdefiniert, um eine VPC-Endpunktrichtlinie anzufügen, die die Berechtigungen steuert, die Prinzipale haben, um Aktionen für Ressourcen über den VPC-Endpunkt durchzuführen. Diese Option ist nur verfügbar, wenn der Service VPC-Endpunktrichtlinien unterstützt. Weitere Informationen finden Sie unter Endpunktrichtlinien.

11. (Optional) Sie fügen ein Tag hinzu, indem Sie neuen Tag hinzufügen auswählen und den Schlüssel und den Wert für den Tag eingeben.
12. Wählen Sie Endpunkt erstellen.

Example – Bucket-Richtlinie für S3 on Outposts

Damit S3 on Outposts mit Ihren VPC-Endpunkten interagieren kann, können Sie dann Ihre S3-on-Outposts-Richtlinie wie folgt aktualisieren:

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3-outposts:*",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

AWS CLI

> **Note**
>
> Um das IPv6-Netzwerk auf Ihrem VPC-Endpunkt zu aktivieren, müssen Sie für den `SupportedIpAddressType` Filter für S3 on Outposts `IPv6` festgelegt haben.

Im folgenden Beispiel wird der `create-vpc-endpoint` Befehl verwendet, um einen neuen Dual-Stack-Schnittstellenendpunkt zu erstellen.

```
aws ec2 create-vpc-endpoint \
--vpc-id vpc-12345678 \
--vpc-endpoint-type Interface \
--service-name com.amazonaws.us-east-1.s3-outposts \
--subnet-id subnet-12345678 \
--security-group-id sg-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

Abhängig von der AWS PrivateLink Servicekonfiguration müssen neu erstellte Endpunktverbindungen möglicherweise vom VPC-Endpunkt-Serviceanbieter akzeptiert werden, bevor sie verwendet werden können. Weitere Informationen finden Sie unter Annehmen und Ablehnen von Endpunktverbindungsanforderungen im AWS PrivateLink -Benutzerhandbuch.

Im folgenden Beispiel wird der `modify-vpc-endpoint` Befehl verwendet, um den reinen IPv-VPC-Endpunkt auf einen Dual-Stack-Endpunkt zu aktualisieren. Der Dual-Stack-Endpunkt ermöglicht den Zugriff sowohl auf das IPv4- als auch auf das IPv6-Netzwerk.

```
aws ec2 modify-vpc-endpoint \
--vpc-endpoint-id vpce-12345678 \
--add-subnet-ids subnet-12345678 \
--remove-subnet-ids subnet-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

Weitere Informationen zum Aktivieren des IPv6-Netzwerks für finden Sie AWS PrivateLinkunter Beschleunigen Sie Ihre IPv6-Einführung mit -AWS PrivateLinkServices und -Endpunkten.

Verwenden von S3-on-Outposts-Dual-Stack-Endpunkten

S3-on-Outposts-Dual-Stack-Endpunkte unterstützen Anfragen an S3-on-Outposts-Buckets über IPv6 und IPv4. In diesem Abschnitt wird beschrieben, wie Sie S3-on-Outposts-Dual-Stack-Endpunkte verwenden.

Themen

- S3-on-Outposts-Dual-Stack-Endpunkte
- Verwenden von Dual-Stack-Endpunkten von der AWS CLI
- Verwenden von S3-on-Outposts-Dual-Stack-Endpunkten über die AWS -SDKs

S3-on-Outposts-Dual-Stack-Endpunkte

Wenn Sie eine Anfrage an einen Dual-Stack-Endpunkt stellen, wird die S3-on-Outposts-Bucket-URL in eine IPv6- oder eine IPv4-Adresse aufgelöst. Weitere Informationen zum Zugriff auf einen S3-on-Outposts-Bucket über IPv6 finden Sie unter Senden von Anforderungen an S3 on Outposts über IPv6.

Um über einen Dual-Stack-Endpunkt auf einen S3-on-Outposts-Bucket zuzugreifen, verwenden Sie einen Endpunktnamen im Pfadformat. S3 on Outposts unterstützt nur regionale Dual-Stack-Endpunktnamen, was bedeutet, dass Sie die Region als Teil des Namens angeben müssen.

Verwenden Sie für einen Dual-Stack-FIPs-Endpunkt im Pfadformat die folgende Namenskonvention:

```
s3-outposts-fips.region.api.aws
```

Verwenden Sie für Dual-Stack-Nicht-FIPS-Endpunkte die folgende Namenskonvention:

```
s3-outposts.region.api.aws
```

> **Note**
>
> Namen von Endpunkten im virtuellen Hosting-Stil werden in S3 on Outposts nicht unterstützt.

Verwenden von Dual-Stack-Endpunkten von der AWS CLI

Dieser Abschnitt enthält Beispiele für AWS CLI-Befehle für Anfragen an einen Dual-Stack-Endpunkt. Weitere Informationen zum Einrichten der AWS CLI finden Sie unter Erste Schritte mit der AWS CLI und dem SDK for Java.

Sie legen den Konfigurationswert `true` in einem Profil in Ihrer -AWS ConfigDatei `use_dualstack_endpoint` auf fest, um alle Amazon S3-Anforderungen von den `s3api` AWS CLI Befehlen `s3` und an den Dual-Stack-Endpunkt für die angegebene Region weiterzuleiten. Sie geben die Region in der Konfigurationsdatei oder in einem Befehl mit der `--region` Option an.

Bei der Verwendung von Dual-Stack-Endpunkten mit der wird AWS CLInur der `path` Adressierungsstil unterstützt. Der in der Konfigurationsdatei festgelegte Adressierungsstil bestimmt, ob der Bucket-Name im Hostnamen oder in der URL enthalten ist. Weitere Informationen finden Sie unter s3outposts im AWS CLI-Benutzerhandbuch.

Um einen Dual-Stack-Endpunkt über die zu verwendenAWS CLI, verwenden Sie den `---endpoint-url`Parameter mit dem - `http://s3.dualstack.region.amazonaws.com` oder -`https://s3-outposts-fips.region.api.aws`Endpunkt für alle `s3control`- oder -`s3outposts`Befehle.

Beispielsweise:

```
$  aws s3control list-regional-buckets --endpoint-url https://s3-
outposts.region.api.aws
```

Verwenden von S3-on-Outposts-Dual-Stack-Endpunkten über die AWS -SDKs

Dieser Abschnitt enthält Beispiele für den Zugriff auf einen Dual-Stack-Endpunkt unter Verwendung der AWS-SDKs.

AWS SDK for Java 2.xBeispiel für einen -Dual-Stack-Endpunkt

Die folgenden Beispiele zeigen, wie Sie die `S3OutpostsClient` Klassen `S3ControlClient` und verwenden, um Dual-Stack-Endpunkte zu aktivieren, wenn Sie einen S3-on-Outposts-Client mit der erstellenAWS SDK for Java 2.x. Anweisungen zum Erstellen und Testen eines funktionierenden Java-Beispiels für Amazon S3 on Outposts finden Sie unter Erste Schritte mit der AWS CLI und dem SDK for Java.

Example – Erstellen einer -**`S3ControlClient`**Klasse mit aktivierten Dual-Stack-Endpunkten

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsRequest;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsResponse;
import software.amazon.awssdk.services.s3control.model.S3ControlException;


public class DualStackEndpointsExample1 {

    public static void main(String[] args) {
        Region clientRegion = Region.of("us-east-1");
        String accountId = "111122223333";
        String navyId = "9876543210";

        try {
            // Create an S3ControlClient with dual-stack endpoints enabled.
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                                                             .region(clientRegion)
                                                             .dualstackEnabled(true)
                                                             .build();
            ListRegionalBucketsRequest listRegionalBucketsRequest =
 ListRegionalBucketsRequest.builder()
```

```
       .accountId(accountId)

       .outpostId(navyId)

       .build();

            ListRegionalBucketsResponse listBuckets =
 s3ControlClient.listRegionalBuckets(listRegionalBucketsRequest);
            System.out.printf("ListRegionalBuckets Response: %s%n",
 listBuckets.toString());
       } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
       }
       catch (S3ControlException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
       } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
       }
   }
}
```

Example – Erstellen eines **`S3OutpostsClient`** mit aktivierten Dual-Stack-Endpunkten

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3outposts.S3OutpostsClient;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsRequest;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsResponse;
import software.amazon.awssdk.services.s3outposts.model.S3OutpostsException;


public class DualStackEndpointsExample2 {

    public static void main(String[] args) {
```

```
        Region clientRegion = Region.of("us-east-1");

        try {
            // Create an S3OutpostsClient with dual-stack endpoints enabled.
            S3OutpostsClient s3OutpostsClient = S3OutpostsClient.builder()
                                                                .region(clientRegion)
                                                                .dualstackEnabled(true)
                                                                .build();
            ListEndpointsRequest listEndpointsRequest =
 ListEndpointsRequest.builder().build();

            ListEndpointsResponse listEndpoints =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
            System.out.printf("ListEndpoints Response: %s%n",
 listEndpoints.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch (S3OutpostsException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
        }
    }
}
```

Wenn Sie die AWS SDK for Java 2.x unter Windows verwenden, müssen Sie möglicherweise die folgende Java Virtual Machine (JVM)-Eigenschaft festlegen:

```
java.net.preferIPv6Addresses=true
```

Codebeispiele für Amazon S3 unter Verwendung von AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie Amazon S3 mit einem AWS -Software Development Kit (SDK) verwenden.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien und serviceübergreifenden Beispiele sehen.

Szenarien sind Codebeispiele, die Ihnen zeigen, wie Sie eine bestimmte Aufgabe ausführen können, indem Sie mehrere Funktionen innerhalb desselben Services aufrufen.

Serviceübergreifende Beispiele sind Beispielanwendungen, die über mehrere AWS-Services hinweg arbeiten.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte

Hello Amazon S3

Die folgenden Codebeispiele veranschaulichen die ersten Schritte mit Amazon S3.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Code für die C MakeLists.txt CMake-Datei.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)
```

```
# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS s3)

# Set this project's name.
project("hello_s3")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_s3.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code für die Quelldatei „hello_s3.cpp“.

```
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <iostream>
#include <aws/core/auth/AWSCredentialsProviderChain.h>
using namespace Aws;
using namespace Aws::Auth;

/*
 *  A "Hello S3" starter application which initializes an Amazon Simple Storage
 Service (Amazon S3) client
 *  and lists the Amazon S3 buckets in the selected region.
 *
 *  main function
 *
 *  Usage: 'hello_s3'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        // You don't normally have to test that you are authenticated. But the
 S3 service permits anonymous requests, thus the s3Client will return "success"
 and 0 buckets even if you are unauthenticated, which can be confusing to a new
 user.
        auto provider =
 Aws::MakeShared<DefaultAWSCredentialsProviderChain>("alloc-tag");
        auto creds = provider->GetAWSCredentials();
        if (creds.IsEmpty()) {
            std::cerr << "Failed authentication" << std::endl;
        }

        Aws::S3::S3Client s3Client(clientConfig);
        auto outcome = s3Client.ListBuckets();
```

```
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed with error: " << outcome.GetError() <<
 std::endl;
            result = 1;
        } else {
            std::cout << "Found " << outcome.GetResult().GetBuckets().size()
                      << " buckets\n";
            for (auto &bucket: outcome.GetResult().GetBuckets()) {
                std::cout << bucket.GetName() << std::endl;
            }
        }
    }

    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for C++ - Referenz für .

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/s3"
```

```
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Storage Service
// (Amazon S3) client and list up to 10 buckets in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 s3Client := s3.NewFromConfig(sdkConfig)
 count := 10
 fmt.Printf("Let's list up to %v buckets for your account.\n", count)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  fmt.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Buckets) == 0 {
  fmt.Println("You don't have any buckets!")
 } else {
  if count > len(result.Buckets) {
   count = len(result.Buckets)
  }
  for _, bucket := range result.Buckets[:count] {
   fmt.Printf("\t%v\n", *bucket.Name)
  }
 }
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for Go - Referenz zu .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Bucket;
import software.amazon.awssdk.services.s3.model.ListBucketsResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloS3 {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBuckets(s3);
    }

    public static void listBuckets(S3Client s3) {
        try {
            ListBucketsResponse response = s3.listBuckets();
            List<Bucket> bucketList = response.buckets();
            bucketList.forEach(bucket -> {
                System.out.println("Bucket Name: " + bucket.name());
```

```
            });

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";

// When no region or credentials are provided, the SDK will use the
// region and credentials from the local AWS config.
const client = new S3Client({});

export const helloS3 = async () => {
  const command = new ListBucketsCommand({});

  const { Buckets } = await client.send(command);
  console.log("Buckets: ");
  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
  return Buckets;
};
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for JavaScript - Referenz für .

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
use Aws\S3\S3Client;

$client = new S3Client(['region' => 'us-west-2']);
$results = $client->listBuckets();
var_dump($results);
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import boto3


def hello_s3():
    """
    Use the AWS SDK for Python (Boto3) to create an Amazon Simple Storage Service
    (Amazon S3) resource and list the buckets in your account.
    This example uses the default settings specified in your shared credentials
    and config files.
```

```
    """
    s3_resource = boto3.resource("s3")
    print("Hello, Amazon S3! Let's list your buckets:")
    for bucket in s3_resource.buckets.all():
        print(f"\t{bucket.name}")


if __name__ == "__main__":
    hello_s3()
```

- Weitere API-Informationen finden Sie unter ListBuckets in der AWS API-Referenz zum -SDK für Python (Boto3).

Codebeispiele

- Aktionen für Amazon S3 mit AWS SDKs
  - Hinzufügen von CORS-Regeln zu einem Amazon S3-Bucket mithilfe eines - AWS SDK
  - Hinzufügen einer Lebenszykluskonfiguration zu einem Amazon S3-Bucket mithilfe eines - AWS SDK
  - Hinzufügen einer Richtlinie zu einem Amazon S3-Bucket mithilfe eines - AWS SDK
  - Abbrechen mehrteiliger Uploads mit einem AWS -SDK
  - Abschließen einer mehrteiligen Upload-Aktion mit einem - AWS SDK
  - Kopieren eines Objekts aus einem Amazon S3-Bucket in einen anderen mithilfe eines - AWS SDK
  - Erstellen eines Amazon S3 Multi-Region Access Points mithilfe eines - AWS SDK
  - Erstellen eines Amazon S3-Buckets mit einem AWS -SDK
  - Erstellen einer mehrteiligen Upload-Struktur mithilfe eines - AWS SDK
  - Löschen von CORS-Regeln aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
  - Löschen einer Richtlinie aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
  - Löschen eines leeren Amazon S3-Buckets mit einem - AWS SDK
  - Löschen eines Amazon S3-Objekts mit einem - AWS SDK
  - Löschen mehrerer Objekte aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
  - Löschen der Lebenszykluskonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
  - Löschen der Website-Konfiguration aus einem Amazon S3-Bucket mithilfe eines - AWS SDK

- Bestimmen des Vorhandenseins und des Inhaltstyps eines Objekts in einem Amazon S3-Bucket mithilfe eines AWS -SDK
- Ermitteln des Vorhandenseins eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Herunterladen aller Objekte aus einem Amazon Simple Storage Service (Amazon S3)-Bucket in ein lokales Verzeichnis
- Aktivieren der Protokollierung für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Aktivieren von Benachrichtigungen für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Aktivieren der Übertragungsbeschleunigung für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- CORS-Regeln für einen Amazon S3-Bucket mit einem - AWS SDK abrufen
- Abrufen eines Amazon S3-Objekts von einem Multi-Region Access Point mithilfe eines - AWS SDK
- Abrufen eines Objekts aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Abrufen eines Objekts aus einem Amazon S3-Bucket mithilfe eines AWS -SDK unter Angabe eines If-Modified-Since-Headers
- Abrufen der ACL eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Abrufen der ACL eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Abrufen der Region, in der sich der Amazon S3-Bucket befindet, mithilfe eines - AWS SDK
- Abrufen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Abrufen der Lebenszykluskonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Abrufen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Abrufen der Richtlinie für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Abrufen der Aufbewahrungskonfiguration eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Abrufen der Website-Konfiguration für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Auflisten von Amazon S3-Buckets mithilfe eines - AWS SDK
- In Bearbeitung befindliche mehrteilige Uploads in einen Amazon S3-Bucket mithilfe eines AWS -SDK auflisten
- Auflisten der Version von Objekten in einem Amazon S3-Bucket mithilfe eines AWS -SDK
- Auflisten von Objekten in einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Wiederherstellen einer archivierten Kopie eines Objekts in einem Amazon S3-Bucket mithilfe eines - AWS SDK

- Festlegen einer neuen ACL für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Festlegen der ACL eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Festlegen des Standardaufbewahrungszeitraums eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Festlegen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Festlegen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Festlegen des Aufbewahrungszeitraums eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Festlegen der Website-Konfiguration für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Beispielansätze für Komponenten- und Integrationstests mit einem - AWS SDK
- Hochladen eines einzelnen Teils eines mehrteiligen Uploads mithilfe eines - AWS SDK
- Hochladen eines Objekts in einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Rekursives Hochladen eines lokalen Verzeichnisses in einen Amazon Simple Storage Service (Amazon S3)-Bucket
- Verwenden von SQL mit Amazon S3 Select zum Abrufen einer Teilmenge von Daten mithilfe eines AWS SDK

- Szenarien für Amazon S3 unter Verwendung von AWS SDKs
  - Erstellen einer vorsignierten URL für Amazon S3 mithilfe eines - AWS SDK
  - Eine Webseite, die Amazon S3-Objekte mithilfe eines -SDK AWS auflistet
  - Erste Schritte mit Amazon S3-Buckets und -Objekten unter Verwendung eines - AWS SDK
  - Erste Schritte mit der Verschlüsselung für Amazon S3-Objekte mithilfe eines - AWS SDK
  - Erste Schritte mit Tags für Amazon S3-Objekte unter Verwendung eines - AWS SDK
  - Arbeiten mit Amazon S3-Objektsperrfunktionen unter Verwendung eines - AWS SDK
  - Verwalten von Zugriffskontrolllisten (ACLs) für Amazon S3-Buckets mithilfe eines - AWS SDK
  - Versionierte Amazon S3-Objekte in Batches mit einer Lambda-Funktion mithilfe eines AWS -SDK verwalten
  - Analysieren von Amazon S3-URIs mit einem AWS -SDK
  - Ausführen einer mehrteiligen Kopie eines Amazon S3-Objekts mithilfe eines - AWS SDK
  - Durchführen eines mehrteiligen Uploads in ein Amazon S3-Objekt mithilfe eines - AWS SDK
  - Hochladen oder Herunterladen großer Dateien in und von Amazon S3 mithilfe eines - AWS SDK

- Hochladen eines Streams unbekannter Größe in ein Amazon S3-Objekt mithilfe eines - AWS SDK
- Verwenden von Prüfsummen für die Arbeit mit einem Amazon S3-Objekt unter Verwendung eines - AWS SDK
- Arbeiten mit versionierten Amazon S3-Objekten unter Verwendung eines - AWS SDK

- Serverless-Beispiele für Amazon S3 unter Verwendung von AWS SDKs
  - Aufrufen einer Lambda-Funktion über einen Amazon-S3-Auslöser
- Serviceübergreifende Beispiele für Amazon S3 unter Verwendung von AWS SDKs
  - Eine Amazon-Transcribe-App entwickeln
  - Konvertieren von Text in Sprache und zurück in Text mithilfe eines - AWS SDK
  - Eine Anwendung für Foto-Asset-Management erstellen, mit der Benutzer Fotos mithilfe von Labels verwalten können
  - Erstellen Sie eine Amazon-Textract-Explorer-Anwendung
  - Erkennen von PSA in Bildern mit Amazon Rekognition mithilfe eines - AWS SDK
  - Erkennen von Entitäten in Text, der mithilfe eines - AWS SDK aus einem Bild extrahiert wurde
  - Erkennen von Gesichtern in einem Bild mithilfe eines - AWS SDK
  - Erkennen von Objekten in Bildern mit Amazon Rekognition mithilfe eines AWS -SDK
  - Erkennen von Personen und Objekten in einem Video mit Amazon Rekognition mithilfe eines AWS -SDK
  - EXIF- und andere Image-Informationen mit einem - AWS SDK speichern

Aktionen für Amazon S3 mit AWS SDKs

Die folgenden Codebeispiele veranschaulichen, wie Sie einzelne Amazon S3-Aktionen mit - AWS SDKs durchführen. Diese Auszüge rufen die Amazon-S3-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der API-Referenz zu Amazon Simple Storage Service (Amazon S3).

Beispiele

- Hinzufügen von CORS-Regeln zu einem Amazon S3-Bucket mithilfe eines - AWS SDK

- Hinzufügen einer Lebenszykluskonfiguration zu einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Hinzufügen einer Richtlinie zu einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Abbrechen mehrteiliger Uploads mit einem AWS -SDK
- Abschließen einer mehrteiligen Upload-Aktion mit einem - AWS SDK
- Kopieren eines Objekts aus einem Amazon S3-Bucket in einen anderen mithilfe eines - AWS SDK
- Erstellen eines Amazon S3 Multi-Region Access Points mithilfe eines - AWS SDK
- Erstellen eines Amazon S3-Buckets mit einem AWS -SDK
- Erstellen einer mehrteiligen Upload-Struktur mithilfe eines - AWS SDK
- Löschen von CORS-Regeln aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Löschen einer Richtlinie aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Löschen eines leeren Amazon S3-Buckets mit einem - AWS SDK
- Löschen eines Amazon S3-Objekts mit einem - AWS SDK
- Löschen mehrerer Objekte aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Löschen der Lebenszykluskonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Löschen der Website-Konfiguration aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Bestimmen des Vorhandenseins und des Inhaltstyps eines Objekts in einem Amazon S3-Bucket mithilfe eines AWS -SDK
- Ermitteln des Vorhandenseins eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Herunterladen aller Objekte aus einem Amazon Simple Storage Service (Amazon S3)-Bucket in ein lokales Verzeichnis
- Aktivieren der Protokollierung für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Aktivieren von Benachrichtigungen für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Aktivieren der Übertragungsbeschleunigung für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- CORS-Regeln für einen Amazon S3-Bucket mit einem - AWS SDK abrufen
- Abrufen eines Amazon S3-Objekts von einem Multi-Region Access Point mithilfe eines - AWS SDK
- Abrufen eines Objekts aus einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Abrufen eines Objekts aus einem Amazon S3-Bucket mithilfe eines AWS -SDK unter Angabe eines If-Modified-Since-Headers
- Abrufen der ACL eines Amazon S3-Buckets mithilfe eines - AWS SDK

- Abrufen der ACL eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Abrufen der Region, in der sich der Amazon S3-Bucket befindet, mithilfe eines - AWS SDK
- Abrufen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Abrufen der Lebenszykluskonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Abrufen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Abrufen der Richtlinie für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Abrufen der Aufbewahrungskonfiguration eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Abrufen der Website-Konfiguration für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Auflisten von Amazon S3-Buckets mithilfe eines - AWS SDK
- In Bearbeitung befindliche mehrteilige Uploads in einen Amazon S3-Bucket mithilfe eines AWS - SDK auflisten
- Auflisten der Version von Objekten in einem Amazon S3-Bucket mithilfe eines AWS -SDK
- Auflisten von Objekten in einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Wiederherstellen einer archivierten Kopie eines Objekts in einem Amazon S3-Bucket mithilfe eines - AWS SDK
- Festlegen einer neuen ACL für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Festlegen der ACL eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Festlegen des Standardaufbewahrungszeitraums eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Festlegen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Festlegen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK
- Festlegen des Aufbewahrungszeitraums eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Festlegen der Website-Konfiguration für einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Beispielansätze für Komponenten- und Integrationstests mit einem - AWS SDK
- Hochladen eines einzelnen Teils eines mehrteiligen Uploads mithilfe eines - AWS SDK
- Hochladen eines Objekts in einen Amazon S3-Bucket mithilfe eines - AWS SDK
- Rekursives Hochladen eines lokalen Verzeichnisses in einen Amazon Simple Storage Service (Amazon S3)-Bucket
- Verwenden von SQL mit Amazon S3 Select zum Abrufen einer Teilmenge von Daten mithilfe eines AWS SDK

Hinzufügen von CORS-Regeln zu einem Amazon S3-Bucket mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie einem S3 Bucket CORS-Regeln (Cross-Origin Resource Sharing) hinzufügen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Add CORS configuration to the Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to apply the CORS configuration to an Amazon S3 bucket.</param>
        /// <param name="configuration">The CORS configuration to apply.</param>
        private static async Task PutCORSConfigurationAsync(AmazonS3Client
client, CORSConfiguration configuration)
        {
            PutCORSConfigurationRequest request = new
PutCORSConfigurationRequest()
            {
                BucketName = BucketName,
                Configuration = configuration,
            };

            _ = await client.PutCORSConfigurationAsync(request);
        }
```

- Weitere API-Informationen finden Sie unter PutBucketCors in der APIAWS SDK for .NET - Referenz für .

CLI

AWS CLI

Im folgenden Beispiel werden PUT-POST, - und -DELETEAnforderungen von www.example.com und -GETAnforderungen von jeder Domain aktiviert:

```
aws s3api put-bucket-cors --bucket MyBucket --cors-configuration file://cors.json

cors.json:
{
  "CORSRules": [
    {
      "AllowedOrigins": ["http://www.example.com"],
      "AllowedHeaders": ["*"],
      "AllowedMethods": ["PUT", "POST", "DELETE"],
      "MaxAgeSeconds": 3000,
      "ExposeHeaders": ["x-amz-server-side-encryption"]
    },
    {
      "AllowedOrigins": ["*"],
      "AllowedHeaders": ["Authorization"],
      "AllowedMethods": ["GET"],
      "MaxAgeSeconds": 3000
    }
  ]
}
```

- API-Details finden Sie unter PutBucketCors in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.s3.S3Client;
import java.util.ArrayList;
import java.util.List;
import software.amazon.awssdk.services.s3.model.GetBucketCorsRequest;
import software.amazon.awssdk.services.s3.model.GetBucketCorsResponse;
import software.amazon.awssdk.services.s3.model.DeleteBucketCorsRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.CORSRule;
import software.amazon.awssdk.services.s3.model.CORSConfiguration;
import software.amazon.awssdk.services.s3.model.PutBucketCorsRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class S3Cors {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <accountId>\s

                Where:
                    bucketName - The Amazon S3 bucket to upload an object into.
                    accountId - The id of the account that owns the Amazon S3
 bucket.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String accountId = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();
```

```
        setCorsInformation(s3, bucketName, accountId);
        getBucketCorsInformation(s3, bucketName, accountId);
        deleteBucketCorsInformation(s3, bucketName, accountId);
        s3.close();
    }

    public static void deleteBucketCorsInformation(S3Client s3, String
bucketName, String accountId) {
        try {
            DeleteBucketCorsRequest bucketCorsRequest =
DeleteBucketCorsRequest.builder()
                    .bucket(bucketName)
                    .expectedBucketOwner(accountId)
                    .build();

            s3.deleteBucketCors(bucketCorsRequest);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getBucketCorsInformation(S3Client s3, String bucketName,
String accountId) {
        try {
            GetBucketCorsRequest bucketCorsRequest =
GetBucketCorsRequest.builder()
                    .bucket(bucketName)
                    .expectedBucketOwner(accountId)
                    .build();

            GetBucketCorsResponse corsResponse =
s3.getBucketCors(bucketCorsRequest);
            List<CORSRule> corsRules = corsResponse.corsRules();
            for (CORSRule rule : corsRules) {
                System.out.println("allowOrigins: " + rule.allowedOrigins());
                System.out.println("AllowedMethod: " + rule.allowedMethods());
            }

        } catch (S3Exception e) {

            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }

    public static void setCorsInformation(S3Client s3, String bucketName, String
 accountId) {
        List<String> allowMethods = new ArrayList<>();
        allowMethods.add("PUT");
        allowMethods.add("POST");
        allowMethods.add("DELETE");

        List<String> allowOrigins = new ArrayList<>();
        allowOrigins.add("http://example.com");
        try {
            // Define CORS rules.
            CORSRule corsRule = CORSRule.builder()
                    .allowedMethods(allowMethods)
                    .allowedOrigins(allowOrigins)
                    .build();

            List<CORSRule> corsRules = new ArrayList<>();
            corsRules.add(corsRule);
            CORSConfiguration configuration = CORSConfiguration.builder()
                    .corsRules(corsRules)
                    .build();

            PutBucketCorsRequest putBucketCorsRequest =
 PutBucketCorsRequest.builder()
                    .bucket(bucketName)
                    .corsConfiguration(configuration)
                    .expectedBucketOwner(accountId)
                    .build();

            s3.putBucketCors(putBucketCorsRequest);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter PutBucketCors in der APIAWS SDK for Java 2.x -Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Fügen Sie eine CORS-Regel hinzu.

```
import { PutBucketCorsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// By default, Amazon S3 doesn't allow cross-origin requests. Use this command
// to explicitly allow cross-origin requests.
export const main = async () => {
  const command = new PutBucketCorsCommand({
    Bucket: "test-bucket",
    CORSConfiguration: {
      CORSRules: [
        {
          // Allow all headers to be sent to this bucket.
          AllowedHeaders: ["*"],
          // Allow only GET and PUT methods to be sent to this bucket.
          AllowedMethods: ["GET", "PUT"],
          // Allow only requests from the specified origin.
          AllowedOrigins: ["https://www.example.com"],
          // Allow the entity tag (ETag) header to be returned in the response.
 The ETag header
          // The entity tag represents a specific version of the object. The ETag
 reflects
          // changes only to the contents of an object, not its metadata.
          ExposeHeaders: ["ETag"],
          // How long the requesting browser should cache the preflight response.
 After
          // this time, the preflight request will have to be made again.
```

```
          MaxAgeSeconds: 3600,
        },
      ],
    },
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter PutBucketCors in der APIAWS SDK for JavaScript -Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name
```

```
    def put_cors(self, cors_rules):
        """
        Apply CORS rules to the bucket. CORS rules specify the HTTP actions that
are
        allowed from other domains.

        :param cors_rules: The CORS rules to apply.
        """
        try:
            self.bucket.Cors().put(CORSConfiguration={"CORSRules": cors_rules})
            logger.info(
                "Put CORS rules %s for bucket '%s'.", cors_rules,
self.bucket.name
            )
        except ClientError:
            logger.exception("Couldn't put CORS rules for bucket %s.",
self.bucket.name)
            raise
```

- Weitere API-Informationen finden Sie unter PutBucketCors in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors
```

```
  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
 an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Sets CORS rules on a bucket.
  #
  # @param allowed_methods [Array<String>] The types of HTTP requests to allow.
  # @param allowed_origins [Array<String>] The origins to allow.
  # @returns [Boolean] True if the CORS rules were set; otherwise, false.
  def set_cors(allowed_methods, allowed_origins)
    @bucket_cors.put(
      cors_configuration: {
        cors_rules: [
          {
            allowed_methods: allowed_methods,
            allowed_origins: allowed_origins,
            allowed_headers: %w[*],
            max_age_seconds: 3600
          }
        ]
      }
    )
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't set CORS rules for #{@bucket_cors.bucket.name}. Here's why:
 #{e.message}"
    false
  end

end
```

- Weitere API-Informationen finden Sie unter PutBucketCors in der APIAWS SDK for Ruby - Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Hinzufügen einer Lebenszykluskonfiguration zu einem Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie einem S3 Bucket eine Lebenszyklus-Konfiguration hinzufügen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Mit versionierten Objekten arbeiten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Adds lifecycle configuration information to the S3 bucket named in
        /// the bucketName parameter.
        /// </summary>
        /// <param name="client">The S3 client used to call the
        /// PutLifecycleConfigurationAsync method.</param>
        /// <param name="bucketName">A string representing the S3 bucket to
        /// which configuration information will be added.</param>
        /// <param name="configuration">A LifecycleConfiguration object that
        /// will be applied to the S3 bucket.</param>
        public static async Task AddExampleLifecycleConfigAsync(IAmazonS3 client,
string bucketName, LifecycleConfiguration configuration)
        {
            var request = new PutLifecycleConfigurationRequest()
            {
                BucketName = bucketName,
                Configuration = configuration,
            };
            var response = await client.PutLifecycleConfigurationAsync(request);
```

```
        }
```

- Weitere API-Informationen finden Sie unter PutBucketLifecycleConfiguration in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

Der folgende Befehl wendet eine Lebenszykluskonfiguration auf einen Bucket mit dem Namen an`my-bucket`:

```
aws s3api put-bucket-lifecycle-configuration --bucket my-bucket --lifecycle-
configuration  file://lifecycle.json
```

Die Datei `lifecycle.json` ist ein JSON-Dokument im aktuellen Ordner, das zwei Regeln angibt:

```
{
    "Rules": [
        {
            "ID": "Move rotated logs to Glacier",
            "Prefix": "rotated/",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Date": "2015-11-10T00:00:00.000Z",
                    "StorageClass": "GLACIER"
                }
            ]
        },
        {
            "Status": "Enabled",
            "Prefix": "",
            "NoncurrentVersionTransitions": [
                {
                    "NoncurrentDays": 2,
                    "StorageClass": "GLACIER"
                }
            ],
```

```
            "ID": "Move old versions to Glacier"
        }
    ]
}
```

Die erste Regel verschiebt Dateien mit dem Präfix am angegebenen Datum `rotated` nach Glacier. Die zweite Regel verschiebt alte Objektversionen nach Glacier, wenn sie nicht mehr aktuell sind. Informationen zu zulässigen Zeitstempelformaten finden Sie unter Angeben von Parameterwerten im AWS CLI-Benutzerhandbuch.

- API-Details finden Sie unter PutBucketLifecycleConfiguration in der AWS CLI - Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.LifecycleRuleFilter;
import software.amazon.awssdk.services.s3.model.Transition;
import
 software.amazon.awssdk.services.s3.model.GetBucketLifecycleConfigurationRequest;
import
 software.amazon.awssdk.services.s3.model.GetBucketLifecycleConfigurationResponse;
import software.amazon.awssdk.services.s3.model.DeleteBucketLifecycleRequest;
import software.amazon.awssdk.services.s3.model.TransitionStorageClass;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.ExpirationStatus;
import software.amazon.awssdk.services.s3.model.BucketLifecycleConfiguration;
import
 software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.ArrayList;
import java.util.List;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class LifecycleConfiguration {
        public static void main(String[] args) {
                final String usage = """

                                Usage:
                                  <bucketName> <accountId>\s

                                Where:
                                  bucketName - The Amazon Simple Storage Service
 (Amazon S3) bucket to upload an object into.
                                  accountId - The id of the account that owns the
 Amazon S3 bucket.
                                """;

                if (args.length != 2) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String bucketName = args[0];
                String accountId = args[1];
                Region region = Region.US_EAST_1;
                S3Client s3 = S3Client.builder()
                                .region(region)
                                .build();

                setLifecycleConfig(s3, bucketName, accountId);
                getLifecycleConfig(s3, bucketName, accountId);
                deleteLifecycleConfig(s3, bucketName, accountId);
                System.out.println("You have successfully created, updated, and
 deleted a Lifecycle configuration");
                s3.close();
        }
```

```
       public static void setLifecycleConfig(S3Client s3, String bucketName,
String accountId) {
               try {
                       // Create a rule to archive objects with the
"glacierobjects/" prefix to Amazon
                       // S3 Glacier.
                       LifecycleRuleFilter ruleFilter =
LifecycleRuleFilter.builder()
                                       .prefix("glacierobjects/")
                                       .build();

                       Transition transition = Transition.builder()

.storageClass(TransitionStorageClass.GLACIER)
                                       .days(0)
                                       .build();

                       LifecycleRule rule1 = LifecycleRule.builder()
                                       .id("Archive immediately rule")
                                       .filter(ruleFilter)
                                       .transitions(transition)
                                       .status(ExpirationStatus.ENABLED)
                                       .build();

                       // Create a second rule.
                       Transition transition2 = Transition.builder()

.storageClass(TransitionStorageClass.GLACIER)
                                       .days(0)
                                       .build();

                       List<Transition> transitionList = new ArrayList<>();
                       transitionList.add(transition2);

                       LifecycleRuleFilter ruleFilter2 =
LifecycleRuleFilter.builder()
                                       .prefix("glacierobjects/")
                                       .build();

                       LifecycleRule rule2 = LifecycleRule.builder()
                                       .id("Archive and then delete rule")
                                       .filter(ruleFilter2)
                                       .transitions(transitionList)
```

```
                                        .status(ExpirationStatus.ENABLED)
                                        .build();

                        // Add the LifecycleRule objects to an ArrayList.
                        ArrayList<LifecycleRule> ruleList = new ArrayList<>();
                        ruleList.add(rule1);
                        ruleList.add(rule2);

                        BucketLifecycleConfiguration lifecycleConfiguration =
BucketLifecycleConfiguration.builder()
                                        .rules(ruleList)
                                        .build();

                        PutBucketLifecycleConfigurationRequest
putBucketLifecycleConfigurationRequest = PutBucketLifecycleConfigurationRequest
                                        .builder()
                                        .bucket(bucketName)

.lifecycleConfiguration(lifecycleConfiguration)
                                        .expectedBucketOwner(accountId)
                                        .build();


s3.putBucketLifecycleConfiguration(putBucketLifecycleConfigurationRequest);

                } catch (S3Exception e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }

        // Retrieve the configuration and add a new rule.
        public static void getLifecycleConfig(S3Client s3, String bucketName,
String accountId) {
                try {
                        GetBucketLifecycleConfigurationRequest
getBucketLifecycleConfigurationRequest = GetBucketLifecycleConfigurationRequest
                                        .builder()
                                        .bucket(bucketName)
                                        .expectedBucketOwner(accountId)
                                        .build();

                        GetBucketLifecycleConfigurationResponse response = s3
```

```
.getBucketLifecycleConfiguration(getBucketLifecycleConfigurationRequest);
                        List<LifecycleRule> newList = new ArrayList<>();
                        List<LifecycleRule> rules = response.rules();
                        for (LifecycleRule rule : rules) {
                                newList.add(rule);
                        }

                        // Add a new rule with both a prefix predicate and a tag
predicate.
                        LifecycleRuleFilter ruleFilter =
LifecycleRuleFilter.builder()
                                        .prefix("YearlyDocuments/")
                                        .build();

                        Transition transition = Transition.builder()

.storageClass(TransitionStorageClass.GLACIER)
                                        .days(3650)
                                        .build();

                        LifecycleRule rule1 = LifecycleRule.builder()
                                        .id("NewRule")
                                        .filter(ruleFilter)
                                        .transitions(transition)
                                        .status(ExpirationStatus.ENABLED)
                                        .build();

                        // Add the new rule to the list.
                        newList.add(rule1);
                        BucketLifecycleConfiguration lifecycleConfiguration =
BucketLifecycleConfiguration.builder()
                                        .rules(newList)
                                        .build();

                        PutBucketLifecycleConfigurationRequest
putBucketLifecycleConfigurationRequest = PutBucketLifecycleConfigurationRequest
                                        .builder()
                                        .bucket(bucketName)

.lifecycleConfiguration(lifecycleConfiguration)
                                        .expectedBucketOwner(accountId)
                                        .build();
```

```
s3.putBucketLifecycleConfiguration(putBucketLifecycleConfigurationRequest);

                } catch (S3Exception e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }

        // Delete the configuration from the Amazon S3 bucket.
        public static void deleteLifecycleConfig(S3Client s3, String bucketName,
String accountId) {
                try {
                        DeleteBucketLifecycleRequest deleteBucketLifecycleRequest
= DeleteBucketLifecycleRequest
                                        .builder()
                                        .bucket(bucketName)
                                        .expectedBucketOwner(accountId)
                                        .build();

                        s3.deleteBucketLifecycle(deleteBucketLifecycleRequest);

                } catch (S3Exception e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

- Weitere API-Informationen finden Sie unter PutBucketLifecycleConfiguration in der APIAWS SDK for Java 2.x -Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def put_lifecycle_configuration(self, lifecycle_rules):
        """
        Apply a lifecycle configuration to the bucket. The lifecycle
 configuration can
        be used to archive or delete the objects in the bucket according to
 specified
        parameters, such as a number of days.

        :param lifecycle_rules: The lifecycle rules to apply.
        """
        try:
            self.bucket.LifecycleConfiguration().put(
                LifecycleConfiguration={"Rules": lifecycle_rules}
            )
            logger.info(
                "Put lifecycle rules %s for bucket '%s'.",
                lifecycle_rules,
                self.bucket.name,
            )
        except ClientError:
            logger.exception(
                "Couldn't put lifecycle rules for bucket '%s'.", self.bucket.name
            )
            raise
```

- Weitere API-Informationen finden Sie unter PutBucketLifecycleConfiguration in der AWS API-Referenz zum -SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Hinzufügen einer Richtlinie zu einem Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie einem S3 Bucket eine Richtlinie hinzufügen.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::PutBucketPolicy(const Aws::String &bucketName,
                                 const Aws::String &policyBody,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    std::shared_ptr<Aws::StringStream> request_body =
            Aws::MakeShared<Aws::StringStream>("");
    *request_body << policyBody;

    Aws::S3::Model::PutBucketPolicyRequest request;
    request.SetBucket(bucketName);
    request.SetBody(request_body);

    Aws::S3::Model::PutBucketPolicyOutcome outcome =
            s3_client.PutBucketPolicy(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutBucketPolicy: "
                  << outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        std::cout << "Set the following policy body for the bucket '" <<
                  bucketName << "':" << std::endl << std::endl;
        std::cout << policyBody << std::endl;
    }

    return outcome.IsSuccess();
}


//! Build a policy JSON string.
/*!
  \sa GetPolicyString()
  \param userArn Aws user Amazon Resource Name (ARN).
      For more information, see https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_identifiers.html#identifiers-arns.
  \param bucketName Name of a bucket.
*/

Aws::String GetPolicyString(const Aws::String &userArn,
                            const Aws::String &bucketName) {
    return
            "{\n"
            "   \"Version\":\"2012-10-17\",\n"
            "   \"Statement\":[\n"
            "       {\n"
            "           \"Sid\": \"1\",\n"
            "           \"Effect\": \"Allow\",\n"
            "           \"Principal\": {\n"
            "               \"AWS\": \""
            + userArn +
            "\"\n""           },\n"
            "           \"Action\": [ \"s3:GetObject\" ],\n"
            "           \"Resource\": [ \"arn:aws:s3:::"
            + bucketName +
            "/*\" ]\n"
            "       }\n"
            "   ]\n"
            "}";
}
```

- Weitere API-Informationen finden Sie unter PutBucketPolicy in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

In diesem Beispiel können alle Benutzer jedes Objekt in abrufen, MyBucket mit Ausnahme der in der MySecretFolder. Außerdem wird dem Root-Benutzer des AWS Kontos die `delete` Berechtigung `put` und erteilt`1234-5678-9012`:

```
aws s3api put-bucket-policy --bucket MyBucket --policy file://policy.json

policy.json:
{
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::MyBucket/*"
      },
      {
         "Effect": "Deny",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::MyBucket/MySecretFolder/*"
      },
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:root"
         },
         "Action": [
            "s3:DeleteObject",
            "s3:PutObject"
         ],
         "Resource": "arn:aws:s3:::MyBucket/*"
      }
   ]
}
```

- API-Details finden Sie unter PutBucketPolicy in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;
import com.fasterxml.jackson.core.JsonParser;
import com.fasterxml.jackson.databind.ObjectMapper;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SetBucketPolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <polFile>

                Where:
                    bucketName - The Amazon S3 bucket to set the policy on.
```

```
                    polFile - A JSON file containing the policy (see the Amazon
S3 Readme for an example).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String polFile = args[1];
        String policyText = getBucketPolicyFromFile(polFile);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setPolicy(s3, bucketName, policyText);
        s3.close();
    }

    public static void setPolicy(S3Client s3, String bucketName, String
policyText) {
        System.out.println("Setting policy:");
        System.out.println("----");
        System.out.println(policyText);
        System.out.println("----");
        System.out.format("On Amazon S3 bucket: \"%s\"\n", bucketName);

        try {
            PutBucketPolicyRequest policyReq = PutBucketPolicyRequest.builder()
                    .bucket(bucketName)
                    .policy(policyText)
                    .build();

            s3.putBucketPolicy(policyReq);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        System.out.println("Done!");
    }
```

```
    // Loads a JSON-formatted policy from a file
    public static String getBucketPolicyFromFile(String policyFile) {

        StringBuilder fileText = new StringBuilder();
        try {
            List<String> lines = Files.readAllLines(Paths.get(policyFile),
 StandardCharsets.UTF_8);
            for (String line : lines) {
                fileText.append(line);
            }

        } catch (IOException e) {
            System.out.format("Problem reading file: \"%s\"", policyFile);
            System.out.println(e.getMessage());
        }

        try {
            final JsonParser parser = new
 ObjectMapper().getFactory().createParser(fileText.toString());
            while (parser.nextToken() != null) {
            }

        } catch (IOException jpe) {
            jpe.printStackTrace();
        }
        return fileText.toString();
    }
}
```

- Weitere API-Informationen finden Sie unter PutBucketPolicy in der APIAWS SDK for Java 2.x -Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Fügen Sie die Richtlinie hinzu.

```
import { PutBucketPolicyCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new PutBucketPolicyCommand({
    Policy: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Sid: "AllowGetObject",
          // Allow this particular user to call GetObject on any object in this
 bucket.
          Effect: "Allow",
          Principal: {
            AWS: "arn:aws:iam::ACCOUNT-ID:user/USERNAME",
          },
          Action: "s3:GetObject",
          Resource: "arn:aws:s3:::BUCKET-NAME/*",
        },
      ],
    }),
    // Apply the preceding policy to this bucket.
    Bucket: "BUCKET-NAME",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
```

```
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter PutBucketPolicy in der APIAWS SDK for JavaScript -Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def put_policy(self, policy):
        """
        Apply a security policy to the bucket. Policies control users' ability
        to perform specific actions, such as listing the objects in the bucket.

        :param policy: The policy to apply to the bucket.
        """
        try:
            self.bucket.Policy().put(Policy=json.dumps(policy))
```

```
            logger.info("Put policy %s for bucket '%s'.", policy,
 self.bucket.name)
        except ClientError:
            logger.exception("Couldn't apply policy to bucket '%s'.",
 self.bucket.name)
            raise
```

- Weitere API-Informationen finden Sie unter PutBucketPolicy in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
 configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Sets a policy on a bucket.
  #
  def set_policy(policy)
    @bucket_policy.put(policy: policy)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't set the policy for #{@bucket_policy.bucket.name}. Here's why:
 #{e.message}"
    false
```

```
  end

end
```

- Weitere API-Informationen finden Sie unter PutBucketPolicy in der APIAWS SDK for Ruby - Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abbrechen mehrteiliger Uploads mit einem AWS -SDK

Das folgende Codebeispiel veranschaulicht, wie Sie eine mehrteilige Uploads abbrechen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Transfer;

/// <summary>
/// This example shows how to use the Amazon Simple Storage Service
/// (Amazon S3) to stop a multi-part upload process using the Amazon S3
/// TransferUtility.
/// </summary>
public class AbortMPU
{
    public static async Task Main()
    {
```

```
            string bucketName = "doc-example-bucket";

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the S3 client object's constructor.
            // For example: RegionEndpoint.USWest2.
            IAmazonS3 client = new AmazonS3Client();

            await AbortMPUAsync(client, bucketName);
        }

        /// <summary>
        /// Cancels the multi-part copy process.
        /// </summary>
        /// <param name="client">The initialized client object used to create
        /// the TransferUtility object.</param>
        /// <param name="bucketName">The name of the S3 bucket where the
        /// multi-part copy operation is in progress.</param>
        public static async Task AbortMPUAsync(IAmazonS3 client, string
bucketName)
        {
            try
            {
                var transferUtility = new TransferUtility(client);

                // Cancel all in-progress uploads initiated before the specified
date.
                await transferUtility.AbortMultipartUploadsAsync(
                    bucketName, DateTime.Now.AddDays(-7));
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter AbortMultipartUploads in der APIAWS SDK for .NET -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abschließen einer mehrteiligen Upload-Aktion mit einem - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie eine mehrteilige Upload-Aktion abschließen.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer mehrteiligen Kopie
- Durchführen eines mehrteiligen Uploads
- Verwenden der Prüfsummen

CLI

AWS CLI

Der folgende Befehl schließt einen mehrteiligen Upload für den Schlüssel `multipart/01` im Bucket ab`my-bucket`:

```
aws s3api complete-multipart-upload --multipart-upload file://
mpustruct --bucket my-bucket --key 'multipart/01' --upload-id
 dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3U
```

Die für diesen Befehl erforderliche Upload-ID wird von ausgegeben `create-multipart-upload` und kann auch mit abgerufen werden`list-multipart-uploads`.

Die Option für den mehrteiligen Upload im obigen Befehl verwendet eine JSON-Struktur, die die Teile des mehrteiligen Uploads beschreibt, die in die vollständige Datei neu zusammengestellt werden sollen. In diesem Beispiel wird das `file://` Präfix verwendet, um die JSON-Struktur aus einer Datei im lokalen Ordner namens zu lade`mpustruct`n.

mpustruct:

```
{
  "Parts": [
    {
      "ETag": "e868e0f4719e394144ef36531ee6824c",
```

```
      "PartNumber": 1
    },
    {
      "ETag": "6bb2b12753d66fe86da4998aa33fffb0",
      "PartNumber": 2
    },
    {
      "ETag": "d0a0112e841abec9c9ec83406f0159c8",
      "PartNumber": 3
    }
  ]
}
```

Der ETag-Wert für jeden Teil wird jedes Mal ausgegeben, wenn Sie einen Teil mit dem `upload-part` Befehl hochladen, und kann auch durch Aufrufen von abgerufen `list-parts` oder berechnet werden, indem die MD5-Prüfsumme jedes Teils verwendet wird.

Ausgabe:

```
{
    "ETag": "\"3944a9f7a4faab7f78788ff6210f63f0-3\"",
    "Bucket": "my-bucket",
    "Location": "https://my-bucket.s3.amazonaws.com/multipart%2F01",
    "Key": "multipart/01"
}
```

- API-Details finden Sie unter CompleteMultipartUpload in der AWS CLI -Befehlsreferenz.

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    let _complete_multipart_upload_res = client
        .complete_multipart_upload()
        .bucket(&bucket_name)
```

```
        .key(&key)
        .multipart_upload(completed_multipart_upload)
        .upload_id(upload_id)
        .send()
        .await
        .unwrap();
```

- Weitere API-Informationen finden Sie unter CompleteMultipartUpload in der API-AWS Referenz zum -SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Kopieren eines Objekts aus einem Amazon S3-Bucket in einen anderen mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie ein S3-Objekt von einem Bucket in einen anderen kopieren.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erste Schritte mit Buckets und Objekten
- Erste Schritte mit der Verschlüsselung

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
```

```
    using Amazon.S3;
    using Amazon.S3.Model;

    public class CopyObject
    {
        public static async Task Main()
        {
            // Specify the AWS Region where your buckets are located if it is
            // different from the AWS Region of the default user.
            IAmazonS3 s3Client = new AmazonS3Client();

            // Remember to change these values to refer to your Amazon S3
 objects.
            string sourceBucketName = "doc-example-bucket1";
            string destinationBucketName = "doc-example-bucket2";
            string sourceObjectKey = "testfile.txt";
            string destinationObjectKey = "testfilecopy.txt";

            Console.WriteLine($"Copying {sourceObjectKey} from {sourceBucketName}
 to ");
            Console.WriteLine($"{destinationBucketName} as
 {destinationObjectKey}");

            var response = await CopyingObjectAsync(
                s3Client,
                sourceObjectKey,
                destinationObjectKey,
                sourceBucketName,
                destinationBucketName);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine("\nCopy complete.");
            }
        }

        /// <summary>
        /// This method calls the AWS SDK for .NET to copy an
        /// object from one Amazon S3 bucket to another.
        /// </summary>
        /// <param name="client">The Amazon S3 client object.</param>
        /// <param name="sourceKey">The name of the object to be copied.</param>
        /// <param name="destinationKey">The name under which to save the copy.</
param>
```

```
        /// <param name="sourceBucketName">The name of the Amazon S3 bucket
        /// where the file is located now.</param>
        /// <param name="destinationBucketName">The name of the Amazon S3
        /// bucket where the copy should be saved.</param>
        /// <returns>Returns a CopyObjectResponse object with the results from
        /// the async call.</returns>
        public static async Task<CopyObjectResponse> CopyingObjectAsync(
            IAmazonS3 client,
            string sourceKey,
            string destinationKey,
            string sourceBucketName,
            string destinationBucketName)
        {
            var response = new CopyObjectResponse();
            try
            {
                var request = new CopyObjectRequest
                {
                    SourceBucket = sourceBucketName,
                    SourceKey = sourceKey,
                    DestinationBucket = destinationBucketName,
                    DestinationKey = destinationKey,
                };
                response = await client.CopyObjectAsync(request);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error copying object: '{ex.Message}'");
            }

            return response;
        }
    }
```

- Weitere API-Informationen finden Sie unter CopyObject in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file from and to.
#       $2 - The key of the source file to copy.
#       $3 - The key of the destination file.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function copy_item_in_bucket() {
  local bucket_name=$1
  local source_key=$2
  local destination_key=$3
  local response

  response=$(aws s3api copy-object \
    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
```

```
    --key "$destination_key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api copy-object operation failed.\n$response"
    return 1
  fi
}
```

- API-Details finden Sie unter CopyObject in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::CopyObject(const Aws::String &objectKey, const Aws::String
 &fromBucket, const Aws::String &toBucket,
                             const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::CopyObjectRequest request;

    request.WithCopySource(fromBucket + "/" + objectKey)
            .WithKey(objectKey)
            .WithBucket(toBucket);

    Aws::S3::Model::CopyObjectOutcome outcome = client.CopyObject(request);
    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: CopyObject: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;

    }
    else {
```

```
        std::cout << "Successfully copied " << objectKey << " from " <<
 fromBucket <<
                  " to " << toBucket << "." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter CopyObject in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl kopiert ein Objekt von `bucket-1` nach `bucket-2`:

```
aws s3api copy-object --copy-source bucket-1/test.txt --key test.txt --bucket
 bucket-2
```

Ausgabe:

```
{
    "CopyObjectResult": {
        "LastModified": "2015-11-10T01:07:25.000Z",
        "ETag": "\"589c8b79c230a6ecd5a7e1d040a9a030\""
    },
    "VersionId": "YdnYvTCVDqRRFA.NFJjy36p0hxifMlkA"
}
```

- API-Details finden Sie unter CopyObject in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// CopyToBucket copies an object in a bucket to another bucket.
func (basics BucketBasics) CopyToBucket(sourceBucket string, destinationBucket
 string, objectKey string) error {
 _, err := basics.S3Client.CopyObject(context.TODO(), &s3.CopyObjectInput{
  Bucket:     aws.String(destinationBucket),
  CopySource: aws.String(fmt.Sprintf("%v/%v", sourceBucket, objectKey)),
  Key:        aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't copy object from %v:%v to %v:%v. Here's why: %v\n",
   sourceBucket, objectKey, destinationBucket, objectKey, err)
 }
 return err
}
```

- Weitere API-Informationen finden Sie unter CopyObject in der APIAWS SDK for Go - Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Kopieren eines Objekts mit einem S3Client.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CopyObjectRequest;
import software.amazon.awssdk.services.s3.model.CopyObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CopyObject {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <objectKey> <fromBucket> <toBucket>

                Where:
                    objectKey - The name of the object (for example, book.pdf).
                    fromBucket - The S3 bucket name that contains the object (for
 example, bucket1).
                    toBucket - The S3 bucket to copy the object to (for example,
 bucket2).
                """;
```

```
        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String objectKey = args[0];
        String fromBucket = args[1];
        String toBucket = args[2];
        System.out.format("Copying object %s from bucket %s to %s\n", objectKey,
 fromBucket, toBucket);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        copyBucketObject(s3, fromBucket, objectKey, toBucket);
        s3.close();
    }

    public static String copyBucketObject(S3Client s3, String fromBucket, String
 objectKey, String toBucket) {
        CopyObjectRequest copyReq = CopyObjectRequest.builder()
                .sourceBucket(fromBucket)
                .sourceKey(objectKey)
                .destinationBucket(toBucket)
                .destinationKey(objectKey)
                .build();

        try {
            CopyObjectResponse copyRes = s3.copyObject(copyReq);
            return copyRes.copyObjectResult().toString();

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

Verwenden Sie einen S3TransferManager, um ein Objekt von einem Bucket in einen anderen zu kopieren. Sehen Sie sich die vollständige Datei an und testen Sie sie.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.model.CopyObjectRequest;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedCopy;
import software.amazon.awssdk.transfer.s3.model.Copy;
import software.amazon.awssdk.transfer.s3.model.CopyRequest;

import java.util.UUID;

    public String copyObject(S3TransferManager transferManager, String
 bucketName,
            String key, String destinationBucket, String destinationKey) {
        CopyObjectRequest copyObjectRequest = CopyObjectRequest.builder()
                .sourceBucket(bucketName)
                .sourceKey(key)
                .destinationBucket(destinationBucket)
                .destinationKey(destinationKey)
                .build();

        CopyRequest copyRequest = CopyRequest.builder()
                .copyObjectRequest(copyObjectRequest)
                .build();

        Copy copy = transferManager.copy(copyRequest);

        CompletedCopy completedCopy = copy.completionFuture().join();
        return completedCopy.response().copyObjectResult().eTag();
    }
```

- Weitere API-Informationen finden Sie unter CopyObject in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Kopieren Sie das Objekt.

```
import { S3Client, CopyObjectCommand } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new CopyObjectCommand({
    CopySource: "SOURCE_BUCKET/SOURCE_OBJECT_KEY",
    Bucket: "DESTINATION_BUCKET",
    Key: "NEW_OBJECT_KEY",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere API-Informationen finden Sie unter CopyObject in der APIAWS SDK for JavaScript - Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun copyBucketObject(
    fromBucket: String,
    objectKey: String,
    toBucket: String
) {
    var encodedUrl = ""
    try {
        encodedUrl = URLEncoder.encode("$fromBucket/$objectKey",
 StandardCharsets.UTF_8.toString())
    } catch (e: UnsupportedEncodingException) {
        println("URL could not be encoded: " + e.message)
    }

    val request = CopyObjectRequest {
        copySource = encodedUrl
        bucket = toBucket
        key = objectKey
    }
    S3Client { region = "us-east-1" }.use { s3 ->
        s3.copyObject(request)
    }
}
```

- Weitere API-Informationen finden Sie unter CopyObject in der API-AWS Referenz zum -SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Einfache Kopie eines Objekts.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $folder = "copied-folder";
            $this->s3client->copyObject([
                'Bucket' => $this->bucketName,
                'CopySource' => "$this->bucketName/$fileName",
                'Key' => "$folder/$fileName-copy",
            ]);
            echo "Copied $fileName to $folder/$fileName-copy.\n";
        } catch (Exception $exception) {
            echo "Failed to copy $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with object copying before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter CopyObject in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def copy(self, dest_object):
        """
        Copies the object to another bucket.

        :param dest_object: The destination object initialized with a bucket and
 key.
                            This is a Boto3 Object resource.
        """
        try:
            dest_object.copy_from(
                CopySource={"Bucket": self.object.bucket_name, "Key":
 self.object.key}
            )
            dest_object.wait_until_exists()
            logger.info(
                "Copied object from %s:%s to %s:%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
        except ClientError:
            logger.exception(
                "Couldn't copy object from %s/%s to %s/%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
            raise
```

- Weitere API-Informationen finden Sie unter CopyObject in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Kopieren Sie ein Objekt.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                        copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket and rename it with the
 target key.
  #
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key)
```

```
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's
 why: #{e.message}"
  end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Kopieren Sie ein Objekt und fügen Sie dem Zielobjekt eine serverseitige Verschlüsselung hinzu.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyEncryptWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                        copy actions.
  def initialize(source_object)
    @source_object = source_object
  end
```

```
  # Copy the source object to the specified target bucket, rename it with the
 target key, and encrypt it.
  #
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key, encryption)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key,
 server_side_encryption: encryption)
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's
 why: #{e.message}"
  end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"
  target_encryption = "AES256"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyEncryptWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key,
 target_encryption)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key} and "\
       "encrypted the target with #{target_object.server_side_encryption}
 encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter CopyObject in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn copy_object(
    client: &Client,
    bucket_name: &str,
    object_key: &str,
    target_key: &str,
) -> Result<CopyObjectOutput, SdkError<CopyObjectError>> {
    let mut source_bucket_and_object: String = "".to_owned();
    source_bucket_and_object.push_str(bucket_name);
    source_bucket_and_object.push('/');
    source_bucket_and_object.push_str(object_key);

    client
        .copy_object()
        .copy_source(source_bucket_and_object)
        .bucket(bucket_name)
        .key(target_key)
        .send()
        .await
}
```

- Weitere API-Informationen finden Sie unter CopyObject in der API-AWS Referenz zum -SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
TRY.
    lo_s3->copyobject(
      iv_bucket = iv_dest_bucket
      iv_key = iv_dest_object
      iv_copysource = |{ iv_src_bucket }/{ iv_src_object }|
    ).
    MESSAGE 'Object copied to another bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
  CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.
```

- Weitere API-Informationen finden Sie unter CopyObject in der AWS API-Referenz zum -SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func copyFile(from sourceBucket: String, name: String, to destBucket:
 String) async throws {
        let srcUrl = ("\(sourceBucket)/
\(name)").addingPercentEncoding(withAllowedCharacters: .urlPathAllowed)

        let input = CopyObjectInput(
            bucket: destBucket,
            copySource: srcUrl,
            key: name
        )
        _ = try await client.copyObject(input: input)
    }
```

- Weitere API-Informationen finden Sie unter CopyObject in der AWS API-Referenz zum -SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen eines Amazon S3 Multi-Region Access Points mithilfe eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie einen Amazon S3 Multi-Region Access Point erstellen.

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Konfigurieren Sie den S3-Steuerungsclient so, dass er eine Anforderung an die Region us-west-2 sendet.

```
        suspend fun createS3ControlClient(): S3ControlClient {
            // Configure your S3ControlClient to send requests to US West
(Oregon).
            val s3Control = S3ControlClient.fromEnvironment {
                region = "us-west-2"
            }
            return s3Control
        }
```

Erstellen Sie den Multi-Region Access Point.

```
    suspend fun createMrap(s3Control: S3ControlClient, accountIdParam: String,
bucketName1: String, bucketName2: String, mrapName: String): String {
        println("Creating MRAP ...")
        val createMrapResponse: CreateMultiRegionAccessPointResponse =
s3Control.createMultiRegionAccessPoint {
            accountId = accountIdParam
            clientToken = UUID.randomUUID().toString()
            details {
                name = mrapName

                regions = listOf(
                    Region {
                        bucket = bucketName1
                    },
                    Region {
                        bucket = bucketName2
                    }
```

```
                )
            }
        }
        val requestToken: String? = createMrapResponse.requestTokenArn

        // Use the request token to check for the status of the
CreateMultiRegionAccessPoint operation.
        if (requestToken != null) {
            waitForSucceededStatus(s3Control, requestToken, accountIdParam)
            println("MRAP created")
        }

        val getMrapResponse = s3Control.getMultiRegionAccessPoint(
            input = GetMultiRegionAccessPointRequest {
                accountId = accountIdParam
                name = mrapName
            }
        )
        val mrapAlias = getMrapResponse.accessPoint?.alias
        return "arn:aws:s3::$accountIdParam:accesspoint/$mrapAlias"
    }
```

Warten Sie, bis der Multi-Region Access Point verfügbar ist.

```
        suspend fun waitForSucceededStatus(s3Control: S3ControlClient,
requestToken: String, accountIdParam: String, timeBetweenChecks: Duration =
1.minutes) {
            var describeResponse: DescribeMultiRegionAccessPointOperationResponse
            describeResponse = s3Control.describeMultiRegionAccessPointOperation(
                input = DescribeMultiRegionAccessPointOperationRequest {
                    accountId = accountIdParam
                    requestTokenArn = requestToken
                }
            )

            var status: String? = describeResponse.asyncOperation?.requestStatus
            while (status != "SUCCEEDED") {
                delay(timeBetweenChecks)
                describeResponse =
s3Control.describeMultiRegionAccessPointOperation(
                    input = DescribeMultiRegionAccessPointOperationRequest {
                        accountId = accountIdParam
```

```
                    requestTokenArn = requestToken
                }
            )
            status = describeResponse.asyncOperation?.requestStatus
            println(status)
        }
    }
```

- Weitere Informationen finden Sie im Entwicklerhandbuch zum AWS SDK für Kotlin.
- Weitere API-Informationen finden Sie unter CreateMultiRegionAccessPoint in der AWS API-Referenz zum -SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen eines Amazon S3-Buckets mit einem AWS -SDK

Die folgenden Codebeispiele zeigen, wie Sie einen S3 Bucket erstellen.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erste Schritte mit Buckets und Objekten
- Mit versionierten Objekten arbeiten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
```

```
        /// Shows how to create a new Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket to create.</param>
        /// <returns>A boolean value representing the success or failure of
        /// the bucket creation process.</returns>
        public static async Task<bool> CreateBucketAsync(IAmazonS3 client, string
bucketName)
        {
            try
            {
                var request = new PutBucketRequest
                {
                    BucketName = bucketName,
                    UseClientRegion = true,
                };

                var response = await client.PutBucketAsync(request);
                return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error creating bucket: '{ex.Message}'");
                return false;
            }
        }
```

Erstellen Sie einen Bucket mit aktivierter Objektsperre.

```
    /// <summary>
    /// Create a new Amazon S3 bucket with object lock actions.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to create.</param>
    /// <param name="enableObjectLock">True to enable object lock on the
bucket.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateBucketWithObjectLock(string bucketName, bool
enableObjectLock)
    {
        Console.WriteLine($"\tCreating bucket {bucketName} with object lock
{enableObjectLock}.");
```

```
        try
        {
            var request = new PutBucketRequest
            {
                BucketName = bucketName,
                UseClientRegion = true,
                ObjectLockEnabledForBucket = enableObjectLock,
            };

            var response = await _amazonS3.PutBucketAsync(request);

            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error creating bucket: '{ex.Message}'");
            return false;
        }
    }
```

- Weitere API-Informationen finden Sie unter CreateBucket in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
#############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
```

```
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

########################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#       -b bucket_name  -- The name of the bucket to create.
#       -r region_code  -- The code for an AWS Region in which to
#                          create the bucket.
#
# Returns:
#       The URL of the bucket that was created.
#     And:
#       0 - If successful.
#       1 - If it fails.
########################################################################
function create_bucket() {
  local bucket_name region_code response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function create_bucket"
    echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
    echo "  -b bucket_name    The name of the bucket. It must be globally
 unique."
    echo "  [-r region_code]    The code for an AWS Region in which the bucket is
 created."
    echo ""
```

```
  }

  # Retrieve the calling parameters.
  while getopts "b:r:h" option; do
    case "${option}" in
      b) bucket_name="${OPTARG}" ;;
      r) region_code="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
  fi

  local bucket_config_arg
  # A location constraint for "us-east-1" returns an error.
  if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
  fi

  iecho "Parameters:\n"
  iecho "    Bucket name:   $bucket_name"
  iecho "    Region code:   $region_code"
  iecho ""

  # If the bucket already exists, we don't want to try to create it.
  if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
  fi

  # shellcheck disable=SC2086
```

```
  response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
  fi
}
```

- API-Details finden Sie unter CreateBucket in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::CreateBucket(const Aws::String &bucketName,
                              const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::CreateBucketRequest request;
    request.SetBucket(bucketName);

    //TODO(user): Change the bucket location constraint enum to your target
 Region.
    if (clientConfig.region != "us-east-1") {
        Aws::S3::Model::CreateBucketConfiguration createBucketConfig;
        createBucketConfig.SetLocationConstraint(

 Aws::S3::Model::BucketLocationConstraintMapper::GetBucketLocationConstraintForName(
                         clientConfig.region));
        request.SetCreateBucketConfiguration(createBucketConfig);
    }
```

```
    Aws::S3::Model::CreateBucketOutcome outcome = client.CreateBucket(request);
    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: CreateBucket: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Created bucket " << bucketName <<
                  " in the specified AWS Region." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter CreateBucket in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Beispiel 1: So erstellen Sie einen Bucket

Im folgenden `create-bucket` Beispiel wird ein Bucket mit dem Namen erstellt`my-bucket`:

```
aws s3api create-bucket \
    --bucket my-bucket \
    --region us-east-1
```

Ausgabe:

```
{
    "Location": "/my-bucket"
}
```

Weitere Informationen finden Sie unter Erstellen eines Buckets im Amazon S3-Benutzerhandbuch.

Beispiel 2: So erstellen Sie einen Bucket mit erzwungenem Eigentümer

Im folgenden `create-bucket` Beispiel wird ein Bucket mit dem Namen erstellt`my-bucket`, der die Einstellung „Bucket-Eigentümer erzwungen“ für S3 Object Ownership verwendet.

```
aws s3api create-bucket \
    --bucket my-bucket \
    --region us-east-1 \
    --object-ownership BucketOwnerEnforced
```

Ausgabe:

```
{
    "Location": "/my-bucket"
}
```

Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs im Amazon-S3-Benutzerhandbuch.

Beispiel 3: So erstellen Sie einen Bucket außerhalb der Region „us-east-1“

Im folgenden `create-bucket` Beispiel wird ein Bucket mit dem Namen `my-bucket` in der `eu-west-1` Region erstellt. Regionen außerhalb von `us-east-1` erfordern `LocationConstraint` die Angabe des entsprechenden , um den Bucket in der gewünschten Region zu erstellen.

```
aws s3api create-bucket \
    --bucket my-bucket \
    --region eu-west-1 \
    --create-bucket-configuration LocationConstraint=eu-west-1
```

Ausgabe:

```
{
    "Location": "http://my-bucket.s3.amazonaws.com/"
}
```

Weitere Informationen finden Sie unter Erstellen eines Buckets im Amazon S3-Benutzerhandbuch.

- API-Details finden Sie unter CreateBucket in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// CreateBucket creates a bucket with the specified name in the specified Region.
func (basics BucketBasics) CreateBucket(name string, region string) error {
 _, err := basics.S3Client.CreateBucket(context.TODO(), &s3.CreateBucketInput{
  Bucket: aws.String(name),
  CreateBucketConfiguration: &types.CreateBucketConfiguration{
   LocationConstraint: types.BucketLocationConstraint(region),
  },
 })
 if err != nil {
  log.Printf("Couldn't create bucket %v in Region %v. Here's why: %v\n",
   name, region, err)
 }
 return err
}
```

- Weitere API-Informationen finden Sie unter CreateBucket in der APIAWS SDK for Go - Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.HeadBucketRequest;
import software.amazon.awssdk.services.s3.model.HeadBucketResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import java.net.URISyntaxException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateBucket {
    public static void main(String[] args) throws URISyntaxException {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The name of the bucket to create. The bucket
 name must be unique, or an error occurs.
                """;
```

```
        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Creating a bucket named %s\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        createBucket(s3, bucketName);
        s3.close();
    }

    public static void createBucket(S3Client s3Client, String bucketName) {
        try {
            S3Waiter s3Waiter = s3Client.waiter();
            CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            s3Client.createBucket(bucketRequest);
            HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            // Wait until the bucket is created and print out the response.
            WaiterResponse<HeadBucketResponse> waiterResponse =
 s3Waiter.waitUntilBucketExists(bucketRequestWait);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println(bucketName + " is ready");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter CreateBucket in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie den Bucket.

```
import { CreateBucketCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new CreateBucketCommand({
    // The name of the bucket. Bucket names are unique and have several other
 constraints.
    // See https://docs.aws.amazon.com/AmazonS3/latest/userguide/
bucketnamingrules.html
    Bucket: "bucket-name",
  });

  try {
    const { Location } = await client.send(command);
    console.log(`Bucket created with location ${Location}`);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter CreateBucket in der APIAWS SDK for JavaScript -Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun createNewBucket(bucketName: String) {
    val request = CreateBucketRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.createBucket(request)
        println("$bucketName is ready")
    }
}
```

- Weitere API-Informationen finden Sie unter CreateBucket in der API-AWS Referenz zum - SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen Bucket.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
```

```
            $this->s3client->createBucket([
                'Bucket' => $this->bucketName,
                'CreateBucketConfiguration' => ['LocationConstraint' => $region],
            ]);
            echo "Created bucket named: $this->bucketName \n";
        } catch (Exception $exception) {
            echo "Failed to create bucket $this->bucketName with error: " .
$exception->getMessage();
            exit("Please fix error with bucket creation before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter CreateBucket in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie einen Bucket mit Standardeinstellungen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def create(self, region_override=None):
        """
```

```
        Create an Amazon S3 bucket in the default Region for the account or in
 the
        specified Region.

        :param region_override: The Region in which to create the bucket. If this
 is
                                not specified, the Region configured in your
 shared
                                credentials is used.
        """
        if region_override is not None:
            region = region_override
        else:
            region = self.bucket.meta.client.meta.region_name
        try:
            self.bucket.create(CreateBucketConfiguration={"LocationConstraint":
 region})

            self.bucket.wait_until_exists()
            logger.info("Created bucket '%s' in region=%s", self.bucket.name,
 region)
        except ClientError as error:
            logger.exception(
                "Couldn't create bucket named '%s' in region=%s.",
                self.bucket.name,
                region,
            )
            raise error
```

Erstellen Sie einen versionierten Bucket mit einer Lebenszyklus-Konfiguration.

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
 lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.
```

```
    Usage is shown in the usage_demo_single_object function at the end of this
module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                   configured lifecycle rules.
    :return: The newly created bucket.
    """
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
expiration},
                    }
                ]
            }
```

```
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
        )
    except ClientError as error:
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket
```

- Weitere API-Informationen finden Sie unter CreateBucket in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketCreateWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An Amazon S3 bucket initialized with a name.
 This is a client-side object until
```

```
  #                                   create is called.
  def initialize(bucket)
    @bucket = bucket
  end

  # Creates an Amazon S3 bucket in the specified AWS Region.
  #
  # @param region [String] The Region where the bucket is created.
  # @return [Boolean] True when the bucket is created; otherwise, false.
  def create?(region)
    @bucket.create(create_bucket_configuration: { location_constraint: region })
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't create bucket. Here's why: #{e.message}"
    false
  end

  # Gets the Region where the bucket is located.
  #
  # @return [String] The location of the bucket.
  def location
    if @bucket.nil?
      "None. You must create a bucket before you can get its location!"
    else
      @bucket.client.get_bucket_location(bucket:
 @bucket.name).location_constraint
    end
  rescue Aws::Errors::ServiceError => e
    "Couldn't get the location of #{@bucket.name}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  region = "us-west-2"
  wrapper = BucketCreateWrapper.new(Aws::S3::Bucket.new("doc-example-bucket-
#{Random.uuid}"))
  return unless wrapper.create?(region)

  puts "Created bucket #{wrapper.bucket.name}."
  puts "Your bucket's region is: #{wrapper.location}"
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter CreateBucket in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn create_bucket(
    client: &Client,
    bucket_name: &str,
    region: &str,
) -> Result<CreateBucketOutput, SdkError<CreateBucketError>> {
    let constraint = BucketLocationConstraint::from(region);
    let cfg = CreateBucketConfiguration::builder()
        .location_constraint(constraint)
        .build();
    client
        .create_bucket()
        .create_bucket_configuration(cfg)
        .bucket(bucket_name)
        .send()
        .await
}
```

- Weitere API-Informationen finden Sie unter CreateBucket in der API-AWS Referenz zum - SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
TRY.
    lo_s3->createbucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'S3 bucket created.' TYPE 'I'.
  CATCH /aws1/cx_s3_bucketalrdyexists.
    MESSAGE 'Bucket name already exists.' TYPE 'E'.
  CATCH /aws1/cx_s3_bktalrdyownedbyyou.
    MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.
```

- Weitere API-Informationen finden Sie unter CreateBucket in der AWS API-Referenz zum -SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func createBucket(name: String) async throws {
        let config = S3ClientTypes.CreateBucketConfiguration(
            locationConstraint: .usEast2
        )
        let input = CreateBucketInput(
            bucket: name,
            createBucketConfiguration: config
        )
        _ = try await client.createBucket(input: input)
    }
```

- Weitere API-Informationen finden Sie unter CreateBucket in der AWS API-Referenz zum - SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen einer mehrteiligen Upload-Struktur mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die -Struktur erstellen, um eine mehrteilige Upload-Aktion zu erstellen.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen einer mehrteiligen Kopie
- Durchführen eines mehrteiligen Uploads
- Verwenden der Prüfsummen

CLI

AWS CLI

Der folgende Befehl erstellt einen mehrteiligen Upload im Bucket `my-bucket` mit dem Schlüssel `multipart/01`:

```
aws s3api create-multipart-upload --bucket my-bucket --key 'multipart/01'
```

Ausgabe:

```
{
    "Bucket": "my-bucket",
    "UploadId":
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3
    "Key": "multipart/01"
}
```

Die abgeschlossene Datei wird `01` in einem Ordner namens `multipart` im Bucket benannt`my-bucket`. Speichern Sie die Upload-ID, den Schlüssel und den Bucket-Namen für die Verwendung mit dem `upload-part` Befehl .

- API-Details finden Sie unter CreateMultipartUpload in der AWS CLI -Befehlsreferenz.

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    let multipart_upload_res: CreateMultipartUploadOutput = client
        .create_multipart_upload()
        .bucket(&bucket_name)
        .key(&key)
        .send()
        .await
        .unwrap();
```

- Weitere API-Informationen finden Sie unter CreateMultipartUpload in der API-AWS Referenz zum -SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Löschen von CORS-Regeln aus einem Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie CORS-Regeln aus einem S3 Bucket löschen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Deletes a CORS configuration from an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to delete the CORS configuration from the bucket.</param>
        private static async Task DeleteCORSConfigurationAsync(AmazonS3Client
client)
        {
            DeleteCORSConfigurationRequest request = new
DeleteCORSConfigurationRequest()
            {
                BucketName = BucketName,
            };
            await client.DeleteCORSConfigurationAsync(request);
        }
```

- Weitere API-Informationen finden Sie unter DeleteBucketCors in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

Der folgende Befehl löscht eine Cross-Origin Resource Sharing-Konfiguration aus einem Bucket namens `my-bucket`:

```
aws s3api delete-bucket-cors --bucket my-bucket
```

- API-Details finden Sie unter DeleteBucketCors in der AWS CLI -Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete_cors(self):
        """
        Delete the CORS rules from the bucket.

        :param bucket_name: The name of the bucket to update.
        """
        try:
```

```
            self.bucket.Cors().delete()
            logger.info("Deleted CORS from bucket '%s'.", self.bucket.name)
        except ClientError:
            logger.exception("Couldn't delete CORS from bucket '%s'.",
 self.bucket.name)
            raise
```

- Weitere API-Informationen finden Sie unter DeleteBucketCors in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
 an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Deletes the CORS configuration of a bucket.
  #
  # @return [Boolean] True if the CORS rules were deleted; otherwise, false.
  def delete_cors
    @bucket_cors.delete
    true
  rescue Aws::Errors::ServiceError => e
```

```
    puts "Couldn't delete CORS rules for #{@bucket_cors.bucket.name}. Here's why:
 #{e.message}"
    false
  end

end
```

- Weitere API-Informationen finden Sie unter DeleteBucketCors in der APIAWS SDK for Ruby -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Löschen einer Richtlinie aus einem Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie eine Richtlinie aus einem S3 Bucket löschen.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::DeleteBucketPolicy(const Aws::String &bucketName,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::DeleteBucketPolicyRequest request;
    request.SetBucket(bucketName);
```

```
    Aws::S3::Model::DeleteBucketPolicyOutcome outcome =
 client.DeleteBucketPolicy(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: DeleteBucketPolicy: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Policy was deleted from the bucket." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter DeleteBucketPolicy in der APIAWS SDK for C++ -Referenz für .

CLI

AWS CLI

Der folgende Befehl löscht eine Bucket-Richtlinie aus einem Bucket mit dem Namen `my-bucket`:

```
aws s3api delete-bucket-policy --bucket my-bucket
```

- API-Details finden Sie unter DeleteBucketPolicy in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketPolicyRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DeleteBucketPolicy {
    public static void main(String[] args) {

        final String usage = """

                Usage:
                    <bucketName>

                Where:
                    bucketName - The Amazon S3 bucket to delete the policy from
 (for example, bucket1).""";

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Deleting policy from bucket: \"%s\"\n\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        deleteS3BucketPolicy(s3, bucketName);
        s3.close();
    }
```

```
    // Delete the bucket policy.
    public static void deleteS3BucketPolicy(S3Client s3, String bucketName) {
        DeleteBucketPolicyRequest delReq = DeleteBucketPolicyRequest.builder()
                .bucket(bucketName)
                .build();

        try {
            s3.deleteBucketPolicy(delReq);
            System.out.println("Done!");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter DeleteBucketPolicy in der APIAWS SDK for Java 2.x -Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie die Bucket-Richtlinie.

```
import { DeleteBucketPolicyCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// This will remove the policy from the bucket.
export const main = async () => {
  const command = new DeleteBucketPolicyCommand({
    Bucket: "test-bucket",
```

```
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter DeleteBucketPolicy in der APIAWS SDK for JavaScript -Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteS3BucketPolicy(bucketName: String?) {
    val request = DeleteBucketPolicyRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteBucketPolicy(request)
        println("Done!")
    }
}
```

- Weitere API-Informationen finden Sie unter DeleteBucketPolicy in der API-AWS Referenz zum -SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete_policy(self):
        """
        Delete the security policy from the bucket.
        """
        try:
            self.bucket.Policy().delete()
            logger.info("Deleted policy for bucket '%s'.", self.bucket.name)
        except ClientError:
            logger.exception(
                "Couldn't delete policy for bucket '%s'.", self.bucket.name
            )
            raise
```

- Weitere API-Informationen finden Sie unter DeleteBucketPolicy in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
 configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  def delete_policy
    @bucket_policy.delete
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't delete the policy from #{@bucket_policy.bucket.name}. Here's
 why: #{e.message}"
    false
  end

end
```

- Weitere API-Informationen finden Sie unter DeleteBucketPolicy in der APIAWS SDK for Ruby -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Löschen eines leeren Amazon S3-Buckets mit einem - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie einen leeren S3 Bucket löschen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Buckets und Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Shows how to delete an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket to
delete.</param>
        /// <returns>A boolean value that represents the success or failure of
        /// the delete operation.</returns>
        public static async Task<bool> DeleteBucketAsync(IAmazonS3 client, string
bucketName)
        {
            var request = new DeleteBucketRequest
            {
                BucketName = bucketName,
            };

            var response = await client.DeleteBucketAsync(request);
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.

# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function delete_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api delete-bucket \
    --bucket "$bucket_name")
```

```
  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
    return 1
  fi
}
```

- API-Details finden Sie unter DeleteBucket in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::DeleteBucket(const Aws::String &bucketName,
                              const Aws::Client::ClientConfiguration
 &clientConfig) {

    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::DeleteBucketRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketOutcome outcome =
            client.DeleteBucket(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: DeleteBucket: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "The bucket was deleted" << std::endl;
    }
```

```
    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl löscht einen Bucket mit dem Namen `my-bucket`:

```
aws s3api delete-bucket --bucket my-bucket --region us-east-1
```

- API-Details finden Sie unter DeleteBucket in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}
```

```
// DeleteBucket deletes a bucket. The bucket must be empty or an error is
 returned.
func (basics BucketBasics) DeleteBucket(bucketName string) error {
 _, err := basics.S3Client.DeleteBucket(context.TODO(), &s3.DeleteBucketInput{
  Bucket: aws.String(bucketName)})
 if err != nil {
  log.Printf("Couldn't delete bucket %v. Here's why: %v\n", bucketName, err)
 }
 return err
}
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der APIAWS SDK for Go - Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
                .bucket(bucket)
                .build();

        s3.deleteBucket(deleteBucketRequest);
        s3.close();
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie den Bucket.

```
import { DeleteBucketCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// Delete a bucket.
export const main = async () => {
  const command = new DeleteBucketCommand({
    Bucket: "test-bucket",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter DeleteBucket in der APIAWS SDK for JavaScript -Referenz für .

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie einen leeren Bucket.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $this->s3client->deleteBucket([
                'Bucket' => $this->bucketName,
            ]);
            echo "Deleted bucket $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to delete $this->bucketName with error: " . $exception-
>getMessage();
            exit("Please fix error with bucket deletion before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
```

```
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete(self):
        """
        Delete the bucket. The bucket must be empty or an error is raised.
        """
        try:
            self.bucket.delete()
            self.bucket.wait_until_not_exists()
            logger.info("Bucket %s successfully deleted.", self.bucket.name)
        except ClientError:
            logger.exception("Couldn't delete bucket %s.", self.bucket.name)
            raise
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der AWS API-Referenz zum - SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Deletes the objects in an Amazon S3 bucket and deletes the bucket.
#
```

```
  # @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
  def delete_bucket(bucket)
    puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?
")
    answer = gets.chomp.downcase
    if answer == "y"
      bucket.objects.batch_delete!
      bucket.delete
      puts("Emptied and deleted bucket #{bucket.name}.\n")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't empty and delete bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_bucket(client: &Client, bucket_name: &str) -> Result<(),
 Error> {
    client.delete_bucket().bucket(bucket_name).send().await?;
    println!("Bucket deleted");
    Ok(())
}
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der API-AWS Referenz zum - SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
TRY.

    lo_s3->deletebucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'Deleted S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der AWS API-Referenz zum - SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func deleteBucket(name: String) async throws {
        let input = DeleteBucketInput(
            bucket: name
        )
        _ = try await client.deleteBucket(input: input)
    }
```

- Weitere API-Informationen finden Sie unter DeleteBucket in der AWS API-Referenz zum - SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Löschen eines Amazon S3-Objekts mit einem - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie ein S3-Objekt löschen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Mit versionierten Objekten arbeiten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie ein Objekt in einem nicht versionierten S3-Bucket.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;
```

```
    /// <summary>
    /// This example shows how to delete an object from a non-versioned Amazon
    /// Simple Storage Service (Amazon S3) bucket.
    /// </summary>
    public class DeleteObject
    {
        /// <summary>
        /// The Main method initializes the necessary variables and then calls
        /// the DeleteObjectNonVersionedBucketAsync method to delete the object
        /// named by the keyName parameter.
        /// </summary>
        public static async Task Main()
        {
            const string bucketName = "doc-example-bucket";
            const string keyName = "testfile.txt";

            // If the Amazon S3 bucket is located in an AWS Region other than the
            // Region of the default account, define the AWS Region for the
            // Amazon S3 bucket in your call to the AmazonS3Client constructor.
            // For example RegionEndpoint.USWest2.
            IAmazonS3 client = new AmazonS3Client();
            await DeleteObjectNonVersionedBucketAsync(client, bucketName,
keyName);
        }

        /// <summary>
        /// The DeleteObjectNonVersionedBucketAsync takes care of deleting the
        /// desired object from the named bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client used to delete
        /// an object from an Amazon S3 bucket.</param>
        /// <param name="bucketName">The name of the bucket from which the
        /// object will be deleted.</param>
        /// <param name="keyName">The name of the object to delete.</param>
        public static async Task DeleteObjectNonVersionedBucketAsync(IAmazonS3
client, string bucketName, string keyName)
        {
            try
            {
                var deleteObjectRequest = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
```

```
                };

                Console.WriteLine($"Deleting object: {keyName}");
                await client.DeleteObjectAsync(deleteObjectRequest);
                Console.WriteLine($"Object: {keyName} deleted from
{bucketName}.");
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error encountered on server.
Message:'{ex.Message}' when deleting an object.");
            }
        }
    }
```

Löschen Sie ein Objekt in einem versionierten S3-Bucket.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example creates an object in an Amazon Simple Storage Service
    /// (Amazon S3) bucket and then deletes the object version that was
    /// created.
    /// </summary>
    public class DeleteObjectVersion
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "verstioned-object.txt";

            // If the AWS Region of the default user is different from the AWS
            // Region of the Amazon S3 bucket, pass the AWS Region of the
            // bucket region to the Amazon S3 client object's constructor.
            // Define it like this:
            //      RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
            IAmazonS3 client = new AmazonS3Client();
```

```
            await CreateAndDeleteObjectVersionAsync(client, bucketName, keyName);
        }

        /// <summary>
        /// This method creates and then deletes a versioned object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
        /// create and delete the object.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
        /// object will be created and deleted.</param>
        /// <param name="keyName">The key name of the object to create.</param>
        public static async Task CreateAndDeleteObjectVersionAsync(IAmazonS3
client, string bucketName, string keyName)
        {
            try
            {
                // Add a sample object.
                string versionID = await PutAnObject(client, bucketName,
keyName);

                // Delete the object by specifying an object key and a version
ID.
                DeleteObjectRequest request = new DeleteObjectRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                    VersionId = versionID,
                };

                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(request);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: {ex.Message}");
            }
        }

        /// <summary>
        /// This method is used to create the temporary Amazon S3 object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 object which will be
used
        /// to create the temporary Amazon S3 object.</param>
```

```
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
 object
        /// will be created.</param>
        /// <param name="objectKey">The name of the Amazon S3 object co create.</
param>
        /// <returns>The Version ID of the created object.</returns>
        public static async Task<string> PutAnObject(IAmazonS3 client, string
 bucketName, string objectKey)
        {
            PutObjectRequest request = new PutObjectRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                ContentBody = "This is the content body!",
            };

            PutObjectResponse response = await client.PutObjectAsync(request);
            return response.VersionId;
        }
    }
```

- Weitere API-Informationen finden Sie unter DeleteObject in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
```

```
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function delete_item_in_bucket
#
# This function deletes the specified file from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - The key (file name) in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_item_in_bucket() {
  local bucket_name=$1
  local key=$2
  local response

  response=$(aws s3api delete-object \
    --bucket "$bucket_name" \
    --key "$key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n
$response"
    return 1
  fi
}
```

- API-Details finden Sie unter DeleteObject in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::DeleteObject(const Aws::String &objectKey,
                              const Aws::String &fromBucket,
                              const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteObjectRequest request;

    request.WithKey(objectKey)
            .WithBucket(fromBucket);

    Aws::S3::Model::DeleteObjectOutcome outcome =
            client.DeleteObject(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: DeleteObject: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully deleted the object." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter DeleteObject in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl löscht ein Objekt mit dem Namen `test.txt` aus einem Bucket mit dem Namen `my-bucket`:

```
aws s3api delete-object --bucket my-bucket --key test.txt
```

Wenn die Bucket-Versionsverwaltung aktiviert ist, enthält die Ausgabe die Versions-ID der Löschmarkierung:

```
{
  "VersionId": "9_gKg5vG56F.TTEUdwkxGpJ3tNDlWlGq",
  "DeleteMarker": true
}
```

Weitere Informationen zum Löschen von Objekten finden Sie unter Löschen von Objekten im Amazon S3-Entwicklerhandbuch.

- API-Details finden Sie unter DeleteObject in der AWS CLI -Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie ein Objekt.

```
import { DeleteObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new DeleteObjectCommand({
```

```
    Bucket: "test-bucket",
    Key: "test-key.txt",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere API-Informationen finden Sie unter DeleteObject in der APIAWS SDK for JavaScript -Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie ein Objekt.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def delete(self):
```

```
        """
        Deletes the object.
        """
        try:
            self.object.delete()
            self.object.wait_until_not_exists()
            logger.info(
                "Deleted object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't delete object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
            raise
```

Setzen Sie ein Objekt auf eine vorherige Version zurück, indem Sie spätere Versionen des Objekts löschen.

```
def rollback_object(bucket, object_key, version_id):
    """
    Rolls back an object to an earlier version by deleting all versions that
    occurred after the specified rollback version.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that holds the object to roll back.
    :param object_key: The object to roll back.
    :param version_id: The version ID to roll back to.
    """
    # Versions must be sorted by last_modified date because delete markers are
    # at the end of the list even when they are interspersed in time.
    versions = sorted(
        bucket.object_versions.filter(Prefix=object_key),
        key=attrgetter("last_modified"),
        reverse=True,
    )
```

```
    logger.debug(
        "Got versions:\n%s",
        "\n".join(
            [
                f"\t{version.version_id}, last modified {version.last_modified}"
                for version in versions
            ]
        ),
    )

    if version_id in [ver.version_id for ver in versions]:
        print(f"Rolling back to version {version_id}")
        for version in versions:
            if version.version_id != version_id:
                version.delete()
                print(f"Deleted version {version.version_id}")
            else:
                break

        print(f"Active version is now {bucket.Object(object_key).version_id}")
    else:
        raise KeyError(
            f"{version_id} was not found in the list of versions for "
 f"{object_key}."
        )
```

Reaktivieren Sie ein gelöschtes Objekt, indem Sie die aktive Löschmarke des Objekts entfernen.

```
def revive_object(bucket, object_key):
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
 marker.
    By removing the delete marker, we make the previous version the latest
 version
    and the object then presents as *not* deleted.
```

```
    Usage is shown in the usage_demo_single_object function at the end of this
module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to revive.
    """
    # Get the latest version for the object.
    response = s3.meta.client.list_object_versions(
        Bucket=bucket.name, Prefix=object_key, MaxKeys=1
    )

    if "DeleteMarkers" in response:
        latest_version = response["DeleteMarkers"][0]
        if latest_version["IsLatest"]:
            logger.info(
                "Object %s was indeed deleted on %s. Let's revive it.",
                object_key,
                latest_version["LastModified"],
            )
            obj = bucket.Object(object_key)
            obj.Version(latest_version["VersionId"]).delete()
            logger.info(
                "Revived %s, active version is now %s  with body '%s'",
                object_key,
                obj.version_id,
                obj.get()["Body"].read(),
            )
        else:
            logger.warning(
                "Delete marker is not the latest version for %s!", object_key
            )
    elif "Versions" in response:
        logger.warning("Got an active version for %s, nothing to do.",
object_key)
    else:
        logger.error("Couldn't get any version info for %s.", object_key)
```

Erstellen Sie einen Lambda-Handler, der eine Löschmarke aus einem S3-Objekt entfernt. Dieser Handler kann verwendet werden, um irrelevante Löschmarkierungen in einem versionierten Bucket effizient zu bereinigen.

```
import logging
from urllib import parse
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
logger.setLevel("INFO")

s3 = boto3.client("s3")


def lambda_handler(event, context):
    """
    Removes a delete marker from the specified versioned object.

    :param event: The S3 batch event that contains the ID of the delete marker
                  to remove.
    :param context: Context about the event.
    :return: A result structure that Amazon S3 uses to interpret the result of
 the
             operation. When the result code is TemporaryFailure, S3 retries the
             operation.
    """
    # Parse job parameters from Amazon S3 batch operations
    invocation_id = event["invocationId"]
    invocation_schema_version = event["invocationSchemaVersion"]

    results = []
    result_code = None
    result_string = None

    task = event["tasks"][0]
    task_id = task["taskId"]

    try:
        obj_key = parse.unquote(task["s3Key"], encoding="utf-8")
        obj_version_id = task["s3VersionId"]
        bucket_name = task["s3BucketArn"].split(":")[-1]

        logger.info(
            "Got task: remove delete marker %s from object %s.", obj_version_id,
 obj_key
        )
```

```
        try:
            # If this call does not raise an error, the object version is not a
 delete
            # marker and should not be deleted.
            response = s3.head_object(
                Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
            )
            result_code = "PermanentFailure"
            result_string = (
                f"Object {obj_key}, ID {obj_version_id} is not " f"a delete
 marker."
            )

            logger.debug(response)
            logger.warning(result_string)
        except ClientError as error:
            delete_marker = error.response["ResponseMetadata"]
["HTTPHeaders"].get(
                "x-amz-delete-marker", "false"
            )
            if delete_marker == "true":
                logger.info(
                    "Object %s, version %s is a delete marker.", obj_key,
 obj_version_id
                )
                try:
                    s3.delete_object(
                        Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
                    )
                    result_code = "Succeeded"
                    result_string = (
                        f"Successfully removed delete marker "
                        f"{obj_version_id} from object {obj_key}."
                    )
                    logger.info(result_string)
                except ClientError as error:
                    # Mark request timeout as a temporary failure so it will be
 retried.
                    if error.response["Error"]["Code"] == "RequestTimeout":
                        result_code = "TemporaryFailure"
                        result_string = (
                            f"Attempt to remove delete marker from  "
                            f"object {obj_key} timed out."
```

```
                    )
                    logger.info(result_string)
                else:
                    raise
        else:
            raise ValueError(
                f"The x-amz-delete-marker header is either not "
                f"present or is not 'true'."
            )
    except Exception as error:
        # Mark all other exceptions as permanent failures.
        result_code = "PermanentFailure"
        result_string = str(error)
        logger.exception(error)
    finally:
        results.append(
            {
                "taskId": task_id,
                "resultCode": result_code,
                "resultString": result_string,
            }
        )
    return {
        "invocationSchemaVersion": invocation_schema_version,
        "treatMissingKeysAs": "PermanentFailure",
        "invocationId": invocation_id,
        "results": results,
    }
```

- Weitere API-Informationen finden Sie unter DeleteObject in der AWS API-Referenz zum - SDK für Python (Boto3).

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn remove_object(client: &Client, bucket: &str, key: &str) -> Result<(),
 Error> {
    client
        .delete_object()
        .bucket(bucket)
        .key(key)
        .send()
        .await?;

    println!("Object deleted.");

    Ok(())
}
```

- Weitere API-Informationen finden Sie unter DeleteObject in der API-AWS Referenz zum - SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    TRY.
        lo_s3->deleteobject(
```

```
            iv_bucket = iv_bucket_name
            iv_key = iv_object_key
        ).
        MESSAGE 'Object deleted from S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Weitere API-Informationen finden Sie unter DeleteObject in der AWS API-Referenz zum - SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public func deleteFile(bucket: String, key: String) async throws {
    let input = DeleteObjectInput(
        bucket: bucket,
        key: key
    )

    do {
        _ = try await client.deleteObject(input: input)
    } catch {
        throw error
    }
}
```

- Weitere API-Informationen finden Sie unter DeleteObject in der API-AWS Referenz zum - SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Löschen mehrerer Objekte aus einem Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird demonstriert, wie Sie mehrere Objekte aus einem S3 Bucket löschen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Buckets und Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie alle Objekte aus einem S3 Bucket.

```
        /// <summary>
        /// Delete all of the objects stored in an existing Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket from which the
        /// contents will be deleted.</param>
```

```
        /// <returns>A boolean value that represents the success or failure of
        /// deleting all of the objects in the bucket.</returns>
        public static async Task<bool> DeleteBucketContentsAsync(IAmazonS3
client, string bucketName)
        {
            // Iterate over the contents of the bucket and delete all objects.
            var request = new ListObjectsV2Request
            {
                BucketName = bucketName,
            };

            try
            {
                ListObjectsV2Response response;

                do
                {
                    response = await client.ListObjectsV2Async(request);
                    response.S3Objects
                        .ForEach(async obj => await
client.DeleteObjectAsync(bucketName, obj.Key));

                    // If the response is truncated, set the request
ContinuationToken
                    // from the NextContinuationToken property of the response.
                    request.ContinuationToken = response.NextContinuationToken;
                }
                while (response.IsTruncated);

                return true;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error deleting objects: {ex.Message}");
                return false;
            }
        }
```

Löschen Sie mehrere Objekte in einem nicht versionierten S3-Bucket.

```
    using System;
```

```
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to delete multiple objects from an Amazon Simple
    /// Storage Service (Amazon S3) bucket.
    /// </summary>
    public class DeleteMultipleObjects
    {
        /// <summary>
        /// The Main method initializes the Amazon S3 client and the name of
        /// the bucket and then passes those values to MultiObjectDeleteAsync.
        /// </summary>
        public static async Task Main()
        {
            const string bucketName = "doc-example-bucket";

            // If the Amazon S3 bucket from which you wish to delete objects is
not
            // located in the same AWS Region as the default user, define the
            // AWS Region for the Amazon S3 bucket as a parameter to the client
            // constructor.
            IAmazonS3 s3Client = new AmazonS3Client();

            await MultiObjectDeleteAsync(s3Client, bucketName);
        }

        /// <summary>
        /// This method uses the passed Amazon S3 client to first create and then
        /// delete three files from the named bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// Amazon S3 methods.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where
objects
        /// will be created and then deleted.</param>
        public static async Task MultiObjectDeleteAsync(IAmazonS3 client, string
bucketName)
        {
            // Create three sample objects which we will then delete.
            var keysAndVersions = await PutObjectsAsync(client, 3, bucketName);
```

```
            // Now perform the multi-object delete, passing the key names and
            // version IDs. Since we are working with a non-versioned bucket,
            // the object keys collection includes null version IDs.
            DeleteObjectsRequest multiObjectDeleteRequest = new
DeleteObjectsRequest
            {
                BucketName = bucketName,
                Objects = keysAndVersions,
            };

            // You can add a specific object key to the delete request using the
            // AddKey method of the multiObjectDeleteRequest.
            try
            {
                DeleteObjectsResponse response = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
            }
            catch (DeleteObjectsException e)
            {
                PrintDeletionErrorStatus(e);
            }
        }

        /// <summary>
        /// Prints the list of errors raised by the call to DeleteObjectsAsync.
        /// </summary>
        /// <param name="ex">A collection of exceptions returned by the call to
        /// DeleteObjectsAsync.</param>
        public static void PrintDeletionErrorStatus(DeleteObjectsException ex)
        {
            DeleteObjectsResponse errorResponse = ex.Response;
            Console.WriteLine("x {0}", errorResponse.DeletedObjects.Count);

            Console.WriteLine($"Successfully deleted
{errorResponse.DeletedObjects.Count}.");
            Console.WriteLine($"No. of objects failed to delete =
{errorResponse.DeleteErrors.Count}");

            Console.WriteLine("Printing error data...");
            foreach (DeleteError deleteError in errorResponse.DeleteErrors)
            {
```

```
                Console.WriteLine($"Object Key:
{deleteError.Key}\t{deleteError.Code}\t{deleteError.Message}");
            }
        }

        /// <summary>
        /// This method creates simple text file objects that can be used in
        /// the delete method.
        /// </summary>
        /// <param name="client">The Amazon S3 client used to call
PutObjectAsync.</param>
        /// <param name="number">The number of objects to create.</param>
        /// <param name="bucketName">The name of the bucket where the objects
        /// will be created.</param>
        /// <returns>A list of keys (object keys) and versions that the calling
        /// method will use to delete the newly created files.</returns>
        public static async Task<List<KeyVersion>> PutObjectsAsync(IAmazonS3
client, int number, string bucketName)
        {
            List<KeyVersion> keys = new List<KeyVersion>();
            for (int i = 0; i < number; i++)
            {
                string key = "ExampleObject-" + new System.Random().Next();
                PutObjectRequest request = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = key,
                    ContentBody = "This is the content body!",
                };

                PutObjectResponse response = await
client.PutObjectAsync(request);

                // For non-versioned bucket operations, we only need the
                // object key.
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = key,
                };
                keys.Add(keyVersion);
            }

            return keys;
        }
```

```
    }
```

Löschen Sie mehrere Objekte in einem versionierten S3-Bucket.

```
    using System;
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to delete objects in a version-enabled Amazon
    /// Simple StorageService (Amazon S3) bucket.
    /// </summary>
    public class DeleteMultipleObjects
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";

            // If the AWS Region for your Amazon S3 bucket is different from
            // the AWS Region of the default user, define the AWS Region for
            // the Amazon S3 bucket and pass it to the client constructor
            // like this:
            // RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
            IAmazonS3 s3Client;

            s3Client = new AmazonS3Client();
            await DeleteMultipleObjectsFromVersionedBucketAsync(s3Client,
bucketName);
        }

        /// <summary>
        /// This method removes multiple versions and objects from a
        /// version-enabled Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
```

```
        /// objects.</param>
        public static async Task
DeleteMultipleObjectsFromVersionedBucketAsync(IAmazonS3 client, string
bucketName)
        {
            // Delete objects (specifying object version in the request).
            await DeleteObjectVersionsAsync(client, bucketName);

            // Delete objects (without specifying object version in the request).
            var deletedObjects = await DeleteObjectsAsync(client, bucketName);

            // Additional exercise - remove the delete markers Amazon S3 returned
from
            // the preceding response. This results in the objects reappearing
            // in the bucket (you can verify the appearance/disappearance of
            // objects in the console).
            await RemoveDeleteMarkersAsync(client, bucketName, deletedObjects);
        }

        /// <summary>
        /// Creates and then deletes non-versioned Amazon S3 objects and then
deletes
        /// them again. The method returns a list of the Amazon S3 objects
deleted.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// PubObjectsAsync and NonVersionedDeleteAsync.</param>
        /// <param name="bucketName">The name of the bucket where the objects
        /// will be created and then deleted.</param>
        /// <returns>A list of DeletedObjects.</returns>
        public static async Task<List<DeletedObject>>
DeleteObjectsAsync(IAmazonS3 client, string bucketName)
        {
            // Upload the sample objects.
            var keysAndVersions2 = await PutObjectsAsync(client, bucketName, 3);

            // Delete objects using only keys. Amazon S3 creates a delete marker
and
            // returns its version ID in the response.
            List<DeletedObject> deletedObjects = await
NonVersionedDeleteAsync(client, bucketName, keysAndVersions2);
            return deletedObjects;
        }
```

```
        /// <summary>
        /// This method creates several temporary objects and then deletes them.
        /// </summary>
        /// <param name="client">The S3 client.</param>
        /// <param name="bucketName">Name of the bucket.</param>
        /// <returns>Async task.</returns>
        public static async Task DeleteObjectVersionsAsync(IAmazonS3 client,
string bucketName)
        {
            // Upload the sample objects.
            var keysAndVersions1 = await PutObjectsAsync(client, bucketName, 3);

            // Delete the specific object versions.
            await VersionedDeleteAsync(client, bucketName, keysAndVersions1);
        }

        /// <summary>
        /// Displays the list of information about deleted files to the console.
        /// </summary>
        /// <param name="e">Error information from the delete process.</param>
        private static void DisplayDeletionErrors(DeleteObjectsException e)
        {
            var errorResponse = e.Response;
            Console.WriteLine($"No. of objects successfully deleted =
{errorResponse.DeletedObjects.Count}");
            Console.WriteLine($"No. of objects failed to delete =
{errorResponse.DeleteErrors.Count}");
            Console.WriteLine("Printing error data...");
            foreach (var deleteError in errorResponse.DeleteErrors)
            {
                Console.WriteLine($"Object Key:
{deleteError.Key}\t{deleteError.Code}\t{deleteError.Message}");
            }
        }

        /// <summary>
        /// Delete multiple objects from a version-enabled bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
```

```
        /// objects.</param>
        /// <param name="keys">A list of key names for the objects to delete.</
param>
        private static async Task VersionedDeleteAsync(IAmazonS3 client, string
 bucketName, List<KeyVersion> keys)
        {
            var multiObjectDeleteRequest = new DeleteObjectsRequest
            {
                BucketName = bucketName,
                Objects = keys, // This includes the object keys and specific
 version IDs.
            };

            try
            {
                Console.WriteLine("Executing VersionedDelete...");
                DeleteObjectsResponse response = await
 client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine($"Successfully deleted all the
 {response.DeletedObjects.Count} items");
            }
            catch (DeleteObjectsException ex)
            {
                DisplayDeletionErrors(ex);
            }
        }

        /// <summary>
        /// Deletes multiple objects from a non-versioned Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
        /// objects.</param>
        /// <param name="keys">A list of key names for the objects to delete.</
param>
        /// <returns>A list of the deleted objects.</returns>
        private static async Task<List<DeletedObject>>
 NonVersionedDeleteAsync(IAmazonS3 client, string bucketName, List<KeyVersion>
 keys)
        {
            // Create a request that includes only the object key names.
```

```
            DeleteObjectsRequest multiObjectDeleteRequest = new
DeleteObjectsRequest();
            multiObjectDeleteRequest.BucketName = bucketName;

            foreach (var key in keys)
            {
                multiObjectDeleteRequest.AddKey(key.Key);
            }

            // Execute DeleteObjectsAsync.
            // The DeleteObjectsAsync method adds a delete marker for each
            // object deleted. You can verify that the objects were removed
            // using the Amazon S3 console.
            DeleteObjectsResponse response;
            try
            {
                Console.WriteLine("Executing NonVersionedDelete...");
                response = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
            }
            catch (DeleteObjectsException ex)
            {
                DisplayDeletionErrors(ex);
                throw; // Some deletions failed. Investigate before continuing.
            }

            // This response contains the DeletedObjects list which we use to
delete the delete markers.
            return response.DeletedObjects;
        }

        /// <summary>
        /// Deletes the markers left after deleting the temporary objects.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
        /// objects.</param>
        /// <param name="deletedObjects">A list of the objects that were
deleted.</param>
```

```
        private static async Task RemoveDeleteMarkersAsync(IAmazonS3 client,
string bucketName, List<DeletedObject> deletedObjects)
        {
            var keyVersionList = new List<KeyVersion>();

            foreach (var deletedObject in deletedObjects)
            {
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = deletedObject.Key,
                    VersionId = deletedObject.DeleteMarkerVersionId,
                };
                keyVersionList.Add(keyVersion);
            }

            // Create another request to delete the delete markers.
            var multiObjectDeleteRequest = new DeleteObjectsRequest
            {
                BucketName = bucketName,
                Objects = keyVersionList,
            };

            // Now, delete the delete marker to bring your objects back to the
bucket.
            try
            {
                Console.WriteLine("Removing the delete markers .....");
                var deleteObjectResponse = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine($"Successfully deleted the
{deleteObjectResponse.DeletedObjects.Count} delete markers");
            }
            catch (DeleteObjectsException ex)
            {
                DisplayDeletionErrors(ex);
            }
        }

        /// <summary>
        /// Create temporary Amazon S3 objects to show how object deletion wors
in an
        /// Amazon S3 bucket with versioning enabled.
        /// </summary>
```

```
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// PutObjectAsync to create temporary objects for the example.</param>
        /// <param name="bucketName">A string representing the name of the S3
        /// bucket where we will create the temporary objects.</param>
        /// <param name="number">The number of temporary objects to create.</
param>
        /// <returns>A list of the KeyVersion objects.</returns>
        private static async Task<List<KeyVersion>> PutObjectsAsync(IAmazonS3
 client, string bucketName, int number)
        {
            var keys = new List<KeyVersion>();

            for (var i = 0; i < number; i++)
            {
                string key = "ObjectToDelete-" + new System.Random().Next();
                PutObjectRequest request = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = key,
                    ContentBody = "This is the content body!",
                };

                var response = await client.PutObjectAsync(request);
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = key,
                    VersionId = response.VersionId,
                };

                keys.Add(keyVersion);
            }

            return keys;
        }
    }
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - A list of keys in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function delete_items_in_bucket() {
  local bucket_name=$1
  local keys=$2
  local response

  # Create the JSON for the items to delete.
  local delete_items
  delete_items="{\"Objects\":["
  for key in $keys; do
    delete_items="$delete_items{\"Key\": \"$key\"},"
```

```
  done
  delete_items=${delete_items%?} # Remove the final comma.
  delete_items="$delete_items]}"

  response=$(aws s3api delete-objects \
    --bucket "$bucket_name" \
    --delete "$delete_items")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n
$response"
    return 1
  fi
}
```

- API-Details finden Sie unter DeleteObjects in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::DeleteObjects(const std::vector<Aws::String> &objectKeys,
                               const Aws::String &fromBucket,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteObjectsRequest request;

    Aws::S3::Model::Delete deleteObject;
    for (const Aws::String& objectKey : objectKeys)
    {

 deleteObject.AddObjects(Aws::S3::Model::ObjectIdentifier().WithKey(objectKey));
    }
```

```
    request.SetDelete(deleteObject);
    request.SetBucket(fromBucket);

    Aws::S3::Model::DeleteObjectsOutcome outcome =
            client.DeleteObjects(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error deleting objects. " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully deleted the objects.";
        for (size_t i = 0; i < objectKeys.size(); ++i)
        {
            std::cout << objectKeys[i];
            if (i < objectKeys.size() - 1)
            {
                std::cout << ", ";
            }
        }

        std::cout << " from bucket " << fromBucket << "." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl löscht ein Objekt aus einem Bucket mit dem Namen `my-bucket`:

```
aws s3api delete-objects --bucket my-bucket --delete file://delete.json
```

`delete.json` ist ein JSON-Dokument im aktuellen Verzeichnis, das das zu löschende Objekt angibt:

```
{
  "Objects": [
    {
      "Key": "test1.txt"
    }
  ],
  "Quiet": false
}
```

Ausgabe:

```
{
    "Deleted": [
        {
            "DeleteMarkerVersionId": "mYAT5Mc6F7aeUL8SS7FAAqUPO1koHwzU",
            "Key": "test1.txt",
            "DeleteMarker": true
        }
    ]
}
```

- API-Details finden Sie unter DeleteObjects in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
```

```
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// DeleteObjects deletes a list of objects from a bucket.
func (basics BucketBasics) DeleteObjects(bucketName string, objectKeys []string)
 error {
 var objectIds []types.ObjectIdentifier
 for _, key := range objectKeys {
  objectIds = append(objectIds, types.ObjectIdentifier{Key: aws.String(key)})
 }
 output, err := basics.S3Client.DeleteObjects(context.TODO(),
 &s3.DeleteObjectsInput{
  Bucket: aws.String(bucketName),
  Delete: &types.Delete{Objects: objectIds},
 })
 if err != nil {
  log.Printf("Couldn't delete objects from bucket %v. Here's why: %v\n",
 bucketName, err)
 } else {
  log.Printf("Deleted %v objects.\n", len(output.Deleted))
 }
 return err
}
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der APIAWS SDK for Go - Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.services.s3.model.Delete;
import software.amazon.awssdk.services.s3.model.DeleteObjectsRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.ArrayList;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DeleteMultiObjects {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <bucketName>

                Where:
                   bucketName - the Amazon S3 bucket name.
                """;

        if (args.length != 1) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        deleteBucketObjects(s3, bucketName);
        s3.close();
    }

    public static void deleteBucketObjects(S3Client s3, String bucketName) {
        // Upload three sample objects to the specfied Amazon S3 bucket.
        ArrayList<ObjectIdentifier> keys = new ArrayList<>();
        PutObjectRequest putOb;
        ObjectIdentifier objectId;

        for (int i = 0; i < 3; i++) {
            String keyName = "delete object example " + i;
            objectId = ObjectIdentifier.builder()
                    .key(keyName)
                    .build();

            putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            s3.putObject(putOb, RequestBody.fromString(keyName));
            keys.add(objectId);
        }

        System.out.println(keys.size() + " objects successfully created.");

        // Delete multiple objects in one request.
        Delete del = Delete.builder()
                .objects(keys)
                .build();

        try {
            DeleteObjectsRequest multiObjectDeleteRequest =
DeleteObjectsRequest.builder()
```

```
                    .bucket(bucketName)
                    .delete(del)
                    .build();

            s3.deleteObjects(multiObjectDeleteRequest);
            System.out.println("Multiple objects are deleted!");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie mehrere Objekte.

```
import { DeleteObjectsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new DeleteObjectsCommand({
    Bucket: "test-bucket",
    Delete: {
      Objects: [{ Key: "object1.txt" }, { Key: "object2.txt" }],
    },
  });
```

```
  try {
    const { Deleted } = await client.send(command);
    console.log(
      `Successfully deleted ${Deleted.length} objects from S3 bucket. Deleted
 objects:`,
    );
    console.log(Deleted.map((d) => ` • ${d.Key}`).join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der APIAWS SDK for JavaScript -Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteBucketObjects(bucketName: String, objectName: String) {
    val objectId = ObjectIdentifier {
        key = objectName
    }

    val delOb = Delete {
        objects = listOf(objectId)
    }

    val request = DeleteObjectsRequest {
        bucket = bucketName
        delete = delOb
    }
```

```
    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteObjects(request)
        println("$objectName was deleted from $bucketName")
    }
}
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der API-AWS Referenz zum - SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie eine Reihe von Objekten aus einer Schlüsselliste.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $objects = [];
            foreach ($contents['Contents'] as $content) {
                $objects[] = [
                    'Key' => $content['Key'],
                ];
            }
            $this->s3client->deleteObjects([
                'Bucket' => $this->bucketName,
                'Delete' => [
                    'Objects' => $objects,
                ],
            ]);
            $check = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            if (count($check) <= 0) {
                throw new Exception("Bucket wasn't empty.");
```

```
            }
            echo "Deleted all objects and folders from $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to delete $fileName from $this->bucketName with error:
 " . $exception->getMessage();
            exit("Please fix error with object deletion before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie eine Reihe von Objekten mithilfe einer Liste von Objektschlüsseln.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    @staticmethod
    def delete_objects(bucket, object_keys):
        """
        Removes a list of objects from a bucket.
        This operation is done as a batch in a single request.
```

```
        :param bucket: The bucket that contains the objects. This is a Boto3
 Bucket
                       resource.
        :param object_keys: The list of keys that identify the objects to remove.
        :return: The response that contains data about which objects were deleted
                 and any that could not be deleted.
        """
        try:
            response = bucket.delete_objects(
                Delete={"Objects": [{"Key": key} for key in object_keys]}
            )
            if "Deleted" in response:
                logger.info(
                    "Deleted objects '%s' from bucket '%s'.",
                    [del_obj["Key"] for del_obj in response["Deleted"]],
                    bucket.name,
                )
            if "Errors" in response:
                logger.warning(
                    "Could not delete objects '%s' from bucket '%s'.",
                    [
                        f"{del_obj['Key']}: {del_obj['Code']}"
                        for del_obj in response["Errors"]
                    ],
                    bucket.name,
                )
        except ClientError:
            logger.exception("Couldn't delete any objects from bucket %s.",
 bucket.name)
            raise
        else:
            return response
```

Löschen Sie alle Objekte im Bucket.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
```

```
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    @staticmethod
    def empty_bucket(bucket):
        """
        Remove all objects from a bucket.

        :param bucket: The bucket to empty. This is a Boto3 Bucket resource.
        """
        try:
            bucket.objects.delete()
            logger.info("Emptied bucket '%s'.", bucket.name)
        except ClientError:
            logger.exception("Couldn't empty bucket '%s'.", bucket.name)
            raise
```

Löschen Sie ein versioniertes Objekt dauerhaft, indem Sie alle seine Versionen löschen.

```
def permanently_delete_object(bucket, object_key):
    """
    Permanently deletes a versioned object by deleting all of its versions.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to delete.
    """
    try:
        bucket.object_versions.filter(Prefix=object_key).delete()
        logger.info("Permanently deleted all versions of object %s.", object_key)
    except ClientError:
        logger.exception("Couldn't delete all versions of %s.", object_key)
        raise
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der AWS API-Referenz zum - SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
 # Deletes the objects in an Amazon S3 bucket and deletes the bucket.
 #
 # @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
 def delete_bucket(bucket)
   puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?
")
   answer = gets.chomp.downcase
   if answer == "y"
     bucket.objects.batch_delete!
     bucket.delete
     puts("Emptied and deleted bucket #{bucket.name}.\n")
   end
 rescue Aws::Errors::ServiceError => e
   puts("Couldn't empty and delete bucket #{bucket.name}.")
   puts("\t#{e.code}: #{e.message}")
   raise
 end
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn delete_objects(client: &Client, bucket_name: &str) ->
 Result<Vec<String>, Error> {
    let objects = client.list_objects_v2().bucket(bucket_name).send().await?;

    let mut delete_objects: Vec<ObjectIdentifier> = vec![];
    for obj in objects.contents() {
        let obj_id = ObjectIdentifier::builder()
            .set_key(Some(obj.key().unwrap().to_string()))
            .build()
            .map_err(Error::from)?;
        delete_objects.push(obj_id);
    }

    let return_keys = delete_objects.iter().map(|o| o.key.clone()).collect();

    if !delete_objects.is_empty() {
        client
            .delete_objects()
            .bucket(bucket_name)
            .delete(
                Delete::builder()
                    .set_objects(Some(delete_objects))
                    .build()
                    .map_err(Error::from)?,
            )
            .send()
            .await?;
    }

    let objects: ListObjectsV2Output =
 client.list_objects_v2().bucket(bucket_name).send().await?;
```

```
    eprintln!("{objects:?}");

    match objects.key_count {
        Some(0) => Ok(return_keys),
        _ => Err(Error::unhandled(
            "There were still objects left in the bucket.",
        )),
    }
}
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der API-AWS Referenz zum - SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func deleteObjects(bucket: String, keys: [String]) async throws {
        let input = DeleteObjectsInput(
            bucket: bucket,
            delete: S3ClientTypes.Delete(
                objects: keys.map({ S3ClientTypes.ObjectIdentifier(key: $0) }),
                quiet: true
            )
        )

        do {
```

```
            let output = try await client.deleteObjects(input: input)

            // As of the last update to this example, any errors are returned
            // in the `output` object's `errors` property. If there are any
            // errors in this array, throw an exception. Once the error
            // handling is finalized in later updates to the AWS SDK for
            // Swift, this example will be updated to handle errors better.

            guard let errors = output.errors else {
                return  // No errors.
            }
            if errors.count != 0 {
                throw ServiceHandlerError.deleteObjectsError
            }
        } catch {
            throw error
        }
    }
```

- Weitere API-Informationen finden Sie unter DeleteObjects in der AWS API-Referenz zum - SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Löschen der Lebenszykluskonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK

Die folgenden Codebeispiele veranschaulichen, wie Sie die Lebenszyklus-Konfiguration eines S3-Buckets löschen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// This method removes the Lifecycle configuration from the named
        /// S3 bucket.
        /// </summary>
        /// <param name="client">The S3 client object used to call
        /// the RemoveLifecycleConfigAsync method.</param>
        /// <param name="bucketName">A string representing the name of the
        /// S3 bucket from which the configuration will be removed.</param>
        public static async Task RemoveLifecycleConfigAsync(IAmazonS3 client,
string bucketName)
        {
            var request = new DeleteLifecycleConfigurationRequest()
            {
                BucketName = bucketName,
            };
            await client.DeleteLifecycleConfigurationAsync(request);
        }
```

- Weitere API-Informationen finden Sie unter DeleteBucketLifecycle in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

Der folgende Befehl löscht eine Lebenszykluskonfiguration aus einem Bucket mit dem Namen `my-bucket`:

```
aws s3api delete-bucket-lifecycle --bucket my-bucket
```

- API-Details finden Sie unter DeleteBucketLifecycle in der AWS CLI -Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete_lifecycle_configuration(self):
        """
        Remove the lifecycle configuration from the specified bucket.
        """
        try:
            self.bucket.LifecycleConfiguration().delete()
            logger.info(
                "Deleted lifecycle configuration for bucket '%s'.",
 self.bucket.name
            )
        except ClientError:
            logger.exception(
                "Couldn't delete lifecycle configuration for bucket '%s'.",
                self.bucket.name,
```

```
            )
            raise
```

- Weitere API-Informationen finden Sie unter DeleteBucketLifecycle in der AWS API-Referenz zum -SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Löschen der Website-Konfiguration aus einem Amazon S3-Bucket mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Website-Konfiguration aus einem S3 Bucket löschen.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::DeleteBucketWebsite(const Aws::String &bucketName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteBucketWebsiteRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketWebsiteOutcome outcome =
            client.DeleteBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
```

```
        std::cerr << "Error: DeleteBucketWebsite: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Website configuration was removed." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter DeleteBucketWebsite in der APIAWS SDK for C ++ -Referenz für .

CLI

AWS CLI

Der folgende Befehl löscht eine Website-Konfiguration aus einem Bucket mit dem Namen `my-bucket`:

```
aws s3api delete-bucket-website --bucket my-bucket
```

- API-Details finden Sie unter DeleteBucketWebsite in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketWebsiteRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DeleteWebsiteConfiguration {
    public static void main(String[] args) {
        final String usage = """

                Usage:     <bucketName>

                Where:
                    bucketName - The Amazon S3 bucket to delete the website
 configuration from.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Deleting website configuration for Amazon S3 bucket:
 %s\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        deleteBucketWebsiteConfig(s3, bucketName);
        System.out.println("Done!");
        s3.close();
    }

    public static void deleteBucketWebsiteConfig(S3Client s3, String bucketName)
 {
        DeleteBucketWebsiteRequest delReq = DeleteBucketWebsiteRequest.builder()
                .bucket(bucketName)
```

```
                .build();

        try {
            s3.deleteBucketWebsite(delReq);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.out.println("Failed to delete website configuration!");
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter DeleteBucketWebsite in der APIAWS SDK for Java 2.x -Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie die Website-Konfiguration aus dem Bucket.

```
import { DeleteBucketWebsiteCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// Disable static website hosting on the bucket.
export const main = async () => {
  const command = new DeleteBucketWebsiteCommand({
    Bucket: "test-bucket",
  });

  try {
    const response = await client.send(command);
    console.log(response);
```

```
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter DeleteBucketWebsite in der APIAWS SDK for JavaScript -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Bestimmen des Vorhandenseins und des Inhaltstyps eines Objekts in einem Amazon S3-Bucket mithilfe eines AWS -SDK

Die folgenden Codebeispiele zeigen, wie Sie das Vorhandensein und den Inhaltstyp eines Objekts in einem S3 Bucket bestimmen.

CLI

AWS CLI

Der folgende Befehl ruft Metadaten für ein Objekt in einem Bucket mit dem Namen abmy-bucket:

```
aws s3api head-object --bucket my-bucket --key index.html
```

Ausgabe:

```
{
    "AcceptRanges": "bytes",
    "ContentType": "text/html",
    "LastModified": "Thu, 16 Apr 2015 18:19:14 GMT",
    "ContentLength": 77,
    "VersionId": "null",
    "ETag": "\"30a6ec7e1a9ad79c203d05a589c8b400\"",
    "Metadata": {}
```

```
}
```

- API-Details finden Sie unter HeadObject in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Bestimmen Sie den Inhaltstyp eines Objekts.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetObjectContentType {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name.\s
                """;
```

```
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getContentType(s3, bucketName, keyName);
        s3.close();
    }

    public static void getContentType(S3Client s3, String bucketName, String
 keyName) {
        try {
            HeadObjectRequest objectRequest = HeadObjectRequest.builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();

            HeadObjectResponse objectHead = s3.headObject(objectRequest);
            String type = objectHead.contentType();
            System.out.println("The object content type is " + type);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Rufen Sie den Wiederherstellungsstatus eines Objekts ab.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
```

```
import software.amazon.awssdk.services.s3.model.S3Exception;

public class GetObjectRestoreStatus {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - A key name that represents the object.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        checkStatus(s3, bucketName, keyName);
        s3.close();
    }

    public static void checkStatus(S3Client s3, String bucketName, String
 keyName) {
        try {
            HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            HeadObjectResponse response = s3.headObject(headObjectRequest);
            System.out.println("The Amazon S3 object restoration status is " +
 response.restore());

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter HeadObject in der APIAWS SDK for Java 2.x - Referenz für .

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectExistsWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Checks whether the object exists.
  #
  # @return [Boolean] True if the object exists; otherwise false.
  def exists?
    @object.exists?
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't check existence of object
 #{@object.bucket.name}:#{@object.key}. Here's why: #{e.message}"
    false
  end
end
```

```
# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectExistsWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  exists = wrapper.exists?

  puts "Object #{object_key} #{exists ? 'does' : 'does not'} exist."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter HeadObject in der APIAWS SDK for Ruby - Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Ermitteln des Vorhandenseins eines Amazon S3-Buckets mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie feststellen, ob ein S3-Bucket vorhanden ist.

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
#############################################################################
# function bucket_exists
#
```

```
# This function checks to see if the specified bucket already exists.
#
# Parameters:
#       $1 - The name of the bucket to check.
#
# Returns:
#       0 - If the bucket already exists.
#       1 - If the bucket doesn't exist.
###############################################################################
function bucket_exists() {
  local bucket_name
  bucket_name=$1

  # Check whether the bucket already exists.
  # We suppress all output - we're interested only in the return code.

  if aws s3api head-bucket \
    --bucket "$bucket_name" \
    >/dev/null 2>&1; then
    return 0 # 0 in Bash script means true.
  else
    return 1 # 1 in Bash script means false.
  fi
}
```

- API-Details finden Sie unter HeadBucket in der AWS CLI -Befehlsreferenz.

CLI

AWS CLI

Der folgende Befehl überprüft den Zugriff auf einen Bucket mit dem Namen `my-bucket`:

```
aws s3api head-bucket --bucket my-bucket
```

Wenn der Bucket vorhanden ist und Sie Zugriff darauf haben, wird keine Ausgabe zurückgegeben. Andernfalls wird eine Fehlermeldung angezeigt. Beispielsweise:

```
A client error (404) occurred when calling the HeadBucket operation: Not Found
```

- API-Details finden Sie unter HeadBucket in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// BucketExists checks whether a bucket exists in the current account.
func (basics BucketBasics) BucketExists(bucketName string) (bool, error) {
 _, err := basics.S3Client.HeadBucket(context.TODO(), &s3.HeadBucketInput{
  Bucket: aws.String(bucketName),
 })
 exists := true
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NotFound:
    log.Printf("Bucket %v is available.\n", bucketName)
    exists = false
    err = nil
   default:
    log.Printf("Either you don't have access to bucket %v or another error
 occurred. "+
     "Here's what happened: %v\n", bucketName, err)
   }
```

```
  }
 } else {
  log.Printf("Bucket %v exists and you already own it.", bucketName)
 }

 return exists, err
}
```

- Weitere API-Informationen finden Sie unter HeadBucket in der APIAWS SDK for Go - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def exists(self):
        """
        Determine whether the bucket exists and you have access to it.

        :return: True when the bucket exists; otherwise, False.
```

```
        """
        try:
            self.bucket.meta.client.head_bucket(Bucket=self.bucket.name)
            logger.info("Bucket %s exists.", self.bucket.name)
            exists = True
        except ClientError:
            logger.warning(
                "Bucket %s doesn't exist or you don't have access to it.",
                self.bucket.name,
            )
            exists = False
        return exists
```

- Weitere API-Informationen finden Sie unter HeadBucket in der AWS API-Referenz zum -SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Herunterladen aller Objekte aus einem Amazon Simple Storage Service (Amazon S3)-Bucket in ein lokales Verzeichnis

Das folgende Codebeispiel zeigt, wie Sie alle Objekte aus einem Amazon Simple Storage Service (Amazon S3)-Bucket in ein lokales Verzeichnis herunterladen.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Verwenden Sie einen S3TransferManager, um alle S3-Objekte in denselben S3-Bucket herunterzuladen. S3 Sehen Sie sich die vollständige Datei an und testen Sie sie.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedDirectoryDownload;
import software.amazon.awssdk.transfer.s3.model.DirectoryDownload;
import software.amazon.awssdk.transfer.s3.model.DownloadDirectoryRequest;

import java.io.IOException;
import java.net.URI;
import java.net.URISyntaxException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.HashSet;
import java.util.Set;
import java.util.UUID;
import java.util.stream.Collectors;

    public Integer downloadObjectsToDirectory(S3TransferManager transferManager,
            URI destinationPathURI, String bucketName) {
        DirectoryDownload directoryDownload =
 transferManager.downloadDirectory(DownloadDirectoryRequest.builder()
                .destination(Paths.get(destinationPathURI))
                .bucket(bucketName)
                .build());
        CompletedDirectoryDownload completedDirectoryDownload =
 directoryDownload.completionFuture().join();

        completedDirectoryDownload.failedTransfers()
                .forEach(fail -> logger.warn("Object [{}] failed to transfer",
 fail.toString()));
        return completedDirectoryDownload.failedTransfers().size();
    }
```

- Weitere API-Informationen finden Sie unter DownloadDirectory in der APIAWS SDK for Java 2.x -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Aktivieren der Protokollierung für einen Amazon S3-Bucket mithilfe eines - AWS SDK

Das folgende Beispiel zeigt, wie Sie die Protokollierung in einem S3-Bucket aktivieren.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

/// <summary>
/// This example shows how to enable logging on an Amazon Simple Storage
/// Service (Amazon S3) bucket. You need to have two Amazon S3 buckets for
/// this example. The first is the bucket for which you wish to enable
/// logging, and the second is the location where you want to store the
/// logs.
/// </summary>
public class ServerAccessLogging
{
    private static IConfiguration _configuration = null!;

    public static async Task Main()
    {
        LoadConfig();
```

```
            string bucketName = _configuration["BucketName"];
            string logBucketName = _configuration["LogBucketName"];
            string logObjectKeyPrefix = _configuration["LogObjectKeyPrefix"];
            string accountId = _configuration["AccountId"];

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the Amazon S3 client object's constructor.
            // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
            IAmazonS3 client = new AmazonS3Client();

            try
            {
                // Update bucket policy for target bucket to allow delivery of
logs to it.
                await SetBucketPolicyToAllowLogDelivery(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix,
                    accountId);

                // Enable logging on the source bucket.
                await EnableLoggingAsync(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }
        }

        /// <summary>
        /// This method grants appropriate permissions for logging to the
        /// Amazon S3 bucket where the logs will be stored.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
used
        /// to apply the bucket policy.</param>
        /// <param name="sourceBucketName">The name of the source bucket.</param>
        /// <param name="logBucketName">The name of the bucket where logging
```

```
        /// information will be stored.</param>
        /// <param name="logPrefix">The logging prefix where the logs should be
delivered.</param>
        /// <param name="accountId">The account id of the account where the
source bucket exists.</param>
        /// <returns>Async task.</returns>
        public static async Task SetBucketPolicyToAllowLogDelivery(
            IAmazonS3 client,
            string sourceBucketName,
            string logBucketName,
            string logPrefix,
            string accountId)
        {
            var resourceArn = @"""arn:aws:s3:::" + logBucketName + "/" +
logPrefix + @"*""";

            var newPolicy = @"{
                                ""Statement"":[{
                                ""Sid"": ""S3ServerAccessLogsPolicy"",
                                ""Effect"": ""Allow"",
                                ""Principal"": { ""Service"":
""logging.s3.amazonaws.com"" },
                                ""Action"": [""s3:PutObject""],
                                ""Resource"": [" + resourceArn + @"],
                                ""Condition"": {
                                ""ArnLike"": { ""aws:SourceArn"":
""arn:aws:s3:::" + sourceBucketName + @""" },
                                ""StringEquals"": { ""aws:SourceAccount"": """ +
accountId + @""" }
                                        }
                                    }]
                                }";
            Console.WriteLine($"The policy to apply to bucket {logBucketName} to
enable logging:");
            Console.WriteLine(newPolicy);

            PutBucketPolicyRequest putRequest = new PutBucketPolicyRequest
            {
                BucketName = logBucketName,
                Policy = newPolicy,
            };
            await client.PutBucketPolicyAsync(putRequest);
            Console.WriteLine("Policy applied.");
        }
```

```
        /// <summary>
        /// This method enables logging for an Amazon S3 bucket. Logs will be
 stored
        /// in the bucket you selected for logging. Selected prefix
        /// will be prepended to each log object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
 used
        /// to configure and apply logging to the selected Amazon S3 bucket.</
param>
        /// <param name="bucketName">The name of the Amazon S3 bucket for which
 you
        /// wish to enable logging.</param>
        /// <param name="logBucketName">The name of the Amazon S3 bucket where
 logging
        /// information will be stored.</param>
        /// <param name="logObjectKeyPrefix">The prefix to prepend to each
        /// object key.</param>
        /// <returns>Async task.</returns>
        public static async Task EnableLoggingAsync(
            IAmazonS3 client,
            string bucketName,
            string logBucketName,
            string logObjectKeyPrefix)
        {
            Console.WriteLine($"Enabling logging for bucket {bucketName}.");
            var loggingConfig = new S3BucketLoggingConfig
            {
                TargetBucketName = logBucketName,
                TargetPrefix = logObjectKeyPrefix,
            };

            var putBucketLoggingRequest = new PutBucketLoggingRequest
            {
                BucketName = bucketName,
                LoggingConfig = loggingConfig,
            };
            await client.PutBucketLoggingAsync(putBucketLoggingRequest);
            Console.WriteLine($"Logging enabled.");
        }

        /// <summary>
        /// Loads configuration from settings files.
```

```
        /// </summary>
        public static void LoadConfig()
        {
            _configuration = new ConfigurationBuilder()
                .SetBasePath(Directory.GetCurrentDirectory())
                .AddJsonFile("settings.json") // Load settings from .json file.
                .AddJsonFile("settings.local.json", true) // Optionally, load
local settings.
                .Build();
        }
    }
```

- Weitere API-Informationen finden Sie unter PutBucketLogging in der APIAWS SDK for .NET -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Aktivieren von Benachrichtigungen für einen Amazon S3-Bucket mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie Benachrichtigungen für einen S3-Bucket aktivieren.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.S3;
```

```
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to enable notifications for an Amazon Simple
    /// Storage Service (Amazon S3) bucket.
    /// </summary>
    public class EnableNotifications
    {
        public static async Task Main()
        {
            const string bucketName = "doc-example-bucket1";
            const string snsTopic = "arn:aws:sns:us-east-2:0123456789ab:bucket-
notify";
            const string sqsQueue = "arn:aws:sqs:us-
east-2:0123456789ab:Example_Queue";

            IAmazonS3 client = new AmazonS3Client(Amazon.RegionEndpoint.USEast2);
            await EnableNotificationAsync(client, bucketName, snsTopic,
 sqsQueue);
        }

        /// <summary>
        /// This method makes the call to the PutBucketNotificationAsync method.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client used to call
        /// the PutBucketNotificationAsync method.</param>
        /// <param name="bucketName">The name of the bucket for which
        /// notifications will be turned on.</param>
        /// <param name="snsTopic">The ARN for the Amazon Simple Notification
        /// Service (Amazon SNS) topic associated with the S3 bucket.</param>
        /// <param name="sqsQueue">The ARN of the Amazon Simple Queue Service
        /// (Amazon SQS) queue to which notifications will be pushed.</param>
        public static async Task EnableNotificationAsync(
            IAmazonS3 client,
            string bucketName,
            string snsTopic,
            string sqsQueue)
        {
            try
            {
                // The bucket for which we are setting up notifications.
                var request = new PutBucketNotificationRequest()
                {
                    BucketName = bucketName,
```

```
                };

                // Defines the topic to use when sending a notification.
                var topicConfig = new TopicConfiguration()
                {
                    Events = new List<EventType> { EventType.ObjectCreatedCopy },
                    Topic = snsTopic,
                };
                request.TopicConfigurations = new List<TopicConfiguration>
                {
                    topicConfig,
                };
                request.QueueConfigurations = new List<QueueConfiguration>
                {
                    new QueueConfiguration()
                    {
                        Events = new List<EventType>
{ EventType.ObjectCreatedPut },
                        Queue = sqsQueue,
                    },
                };

                // Now apply the notification settings to the bucket.
                PutBucketNotificationResponse response = await
client.PutBucketNotificationAsync(request);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: {ex.Message}");
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter PutBucketNotificationConfiguration in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

Der wendet eine Benachrichtigungskonfiguration auf einen Bucket mit dem Namen an`my-bucket`:

```
aws s3api put-bucket-notification --bucket my-bucket --notification-configuration
 file://notification.json
```

Die Datei `notification.json` ist ein JSON-Dokument im aktuellen Ordner, das ein SNS-Thema und einen zu überwachenden Ereignistyp angibt:

```
{
  "TopicConfiguration": {
    "Event": "s3:ObjectCreated:*",
    "Topic": "arn:aws:sns:us-west-2:123456789012:s3-notification-topic"
  }
}
```

Dem SNS-Thema muss eine IAM-Richtlinie zugeordnet sein, die es Amazon S3 ermöglicht, darin zu veröffentlichen:

```
{
 "Version": "2008-10-17",
 "Id": "example-ID",
 "Statement": [
  {
   "Sid": "example-statement-ID",
   "Effect": "Allow",
   "Principal": {
     "Service": "s3.amazonaws.com"
   },
   "Action": [
    "SNS:Publish"
   ],
   "Resource": "arn:aws:sns:us-west-2:123456789012:my-bucket",
   "Condition": {
      "ArnLike": {
      "aws:SourceArn": "arn:aws:s3:*:*:my-bucket"
    }
   }
```

```
  }
 ]
}
```

- API-Details finden Sie unter PutBucketNotificationConfiguration in der AWS CLI - Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Event;
import software.amazon.awssdk.services.s3.model.NotificationConfiguration;
import
 software.amazon.awssdk.services.s3.model.PutBucketNotificationConfigurationRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.TopicConfiguration;
import java.util.ArrayList;
import java.util.List;

public class SetBucketEventBridgeNotification {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket.\s
                    topicArn - The Simple Notification Service topic ARN.\s
                    id - An id value used for the topic configuration. This value
 is displayed in the AWS Management Console.\s
                """;
```

```
        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String topicArn = args[1];
        String id = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3Client = S3Client.builder()
                .region(region)
                .build();

        setBucketNotification(s3Client, bucketName, topicArn, id);
        s3Client.close();
    }

    public static void setBucketNotification(S3Client s3Client, String
bucketName, String topicArn, String id) {
        try {
            List<Event> events = new ArrayList<>();
            events.add(Event.S3_OBJECT_CREATED_PUT);

            TopicConfiguration config = TopicConfiguration.builder()
                    .topicArn(topicArn)
                    .events(events)
                    .id(id)
                    .build();

            List<TopicConfiguration> topics = new ArrayList<>();
            topics.add(config);

            NotificationConfiguration configuration =
NotificationConfiguration.builder()
                    .topicConfigurations(topics)
                    .build();

            PutBucketNotificationConfigurationRequest configurationRequest =
PutBucketNotificationConfigurationRequest
                    .builder()
                    .bucket(bucketName)
                    .notificationConfiguration(configuration)
                    .skipDestinationValidation(true)
```

```
                    .build();

            // Set the bucket notification configuration.
            s3Client.putBucketNotificationConfiguration(configurationRequest);
            System.out.println("Added bucket " + bucketName + " with EventBridge
 events enabled.");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter PutBucketNotificationConfiguration in der APIAWS SDK for Java 2.x -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Aktivieren der Übertragungsbeschleunigung für einen Amazon S3-Bucket mithilfe eines - AWS SDK

Das folgende Beispiel zeigt, wie Sie Transfer Acceleration für einen S3-Bucket aktivieren.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
```

```
    using Amazon.S3.Model;

    /// <summary>
    /// Amazon Simple Storage Service (Amazon S3) Transfer Acceleration is a
    /// bucket-level feature that enables you to perform faster data transfers
    /// to Amazon S3. This example shows how to configure Transfer
    /// Acceleration.
    /// </summary>
    public class TransferAcceleration
    {
        /// <summary>
        /// The main method initializes the client object and sets the
        /// Amazon Simple Storage Service (Amazon S3) bucket name before
        /// calling EnableAccelerationAsync.
        /// </summary>
        public static async Task Main()
        {
            var s3Client = new AmazonS3Client();
            const string bucketName = "doc-example-bucket";

            await EnableAccelerationAsync(s3Client, bucketName);
        }

        /// <summary>
        /// This method sets the configuration to enable transfer acceleration
        /// for the bucket referred to in the bucketName parameter.
        /// </summary>
        /// <param name="client">An Amazon S3 client used to enable the
        /// acceleration on an Amazon S3 bucket.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket for which
the
        /// method will be enabling acceleration.</param>
        private static async Task EnableAccelerationAsync(AmazonS3Client client,
string bucketName)
        {
            try
            {
                var putRequest = new PutBucketAccelerateConfigurationRequest
                {
                    BucketName = bucketName,
                    AccelerateConfiguration = new AccelerateConfiguration
                    {
                        Status = BucketAccelerateStatus.Enabled,
                    },
```

```
                };
                await client.PutBucketAccelerateConfigurationAsync(putRequest);

                var getRequest = new GetBucketAccelerateConfigurationRequest
                {
                    BucketName = bucketName,
                };
                var response = await
client.GetBucketAccelerateConfigurationAsync(getRequest);

                Console.WriteLine($"Acceleration state = '{response.Status}' ");
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error occurred. Message:'{ex.Message}' when
setting transfer acceleration");
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter PutBucketAccelerateConfiguration in der APIAWS SDK for .NET -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

CORS-Regeln für einen Amazon S3-Bucket mit einem - AWS SDK abrufen

Die folgenden Codebeispiele zeigen, wie CORS-Regeln (Cross-Origin Resource Sharing) für einen S3 Bucket abgerufen werden.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Retrieve the CORS configuration applied to the Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to retrieve the CORS configuration.</param>
        /// <returns>The created CORS configuration object.</returns>
        private static async Task<CORSConfiguration>
RetrieveCORSConfigurationAsync(AmazonS3Client client)
        {
            GetCORSConfigurationRequest request = new
GetCORSConfigurationRequest()
            {
                BucketName = BucketName,
            };
            var response = await client.GetCORSConfigurationAsync(request);
            var configuration = response.Configuration;
            PrintCORSRules(configuration);
            return configuration;
        }
```

- Weitere API-Informationen finden Sie unter GetBucketCors in der APIAWS SDK for .NET - Referenz für .

CLI

AWS CLI

Der folgende Befehl ruft die Cross-Origin Resource Sharing-Konfiguration für einen Bucket mit dem Namen ab`my-bucket`:

```
aws s3api get-bucket-cors --bucket my-bucket
```

Ausgabe:

```
{
    "CORSRules": [
        {
            "AllowedHeaders": [
                "*"
            ],
            "ExposeHeaders": [
                "x-amz-server-side-encryption"
            ],
            "AllowedMethods": [
                "PUT",
                "POST",
                "DELETE"
            ],
            "MaxAgeSeconds": 3000,
            "AllowedOrigins": [
                "http://www.example.com"
            ]
        },
        {
            "AllowedHeaders": [
                "Authorization"
            ],
            "MaxAgeSeconds": 3000,
            "AllowedMethods": [
                "GET"
            ],
            "AllowedOrigins": [
                "*"
            ]
        }
    ]
```

```
}
```

- API-Details finden Sie unter GetBucketCors in der AWS CLI -Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die CORS-Richtlinie für den Bucket ab.

```
import { GetBucketCorsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new GetBucketCorsCommand({
    Bucket: "test-bucket",
  });

  try {
    const { CORSRules } = await client.send(command);
    CORSRules.forEach((cr, i) => {
      console.log(
        `\nCORSRule ${i + 1}`,
        `\n${"-".repeat(10)}`,
        `\nAllowedHeaders: ${cr.AllowedHeaders.join(" ")}`,
        `\nAllowedMethods: ${cr.AllowedMethods.join(" ")}`,
        `\nAllowedOrigins: ${cr.AllowedOrigins.join(" ")}`,
        `\nExposeHeaders: ${cr.ExposeHeaders.join(" ")}`,
        `\nMaxAgeSeconds: ${cr.MaxAgeSeconds}`,
      );
    });
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter GetBucketCors in der APIAWS SDK for JavaScript -Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def get_cors(self):
        """
        Get the CORS rules for the bucket.

        :return The CORS rules for the specified bucket.
        """
        try:
            cors = self.bucket.Cors()
            logger.info(
                "Got CORS rules %s for bucket '%s'.", cors.cors_rules,
 self.bucket.name
            )
```

```
        except ClientError:
            logger.exception(("Couldn't get CORS for bucket %s.",
 self.bucket.name))
            raise
        else:
            return cors
```

- Weitere API-Informationen finden Sie unter GetBucketCors in der AWS API-Referenz zum - SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
 an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Gets the CORS configuration of a bucket.
  #
  # @return [Aws::S3::Type::GetBucketCorsOutput, nil] The current CORS
 configuration for the bucket.
  def get_cors
    @bucket_cors.data
  rescue Aws::Errors::ServiceError => e
```

```
    puts "Couldn't get CORS configuration for #{@bucket_cors.bucket.name}. Here's
 why: #{e.message}"
    nil
  end

end
```

- Weitere API-Informationen finden Sie unter GetBucketCors in der APIAWS SDK for Ruby - Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen eines Amazon S3-Objekts von einem Multi-Region Access Point mithilfe eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie ein Objekt von einem Multi-Region Access Point abrufen.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erste Schritte mit Buckets und Objekten
- Erste Schritte mit der Verschlüsselung

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Konfigurieren Sie den S3-Client für die Verwendung des Asymmetrischen Sigv4 (Sigv4a)-Signaturalgorithmus.

```
        suspend fun createS3Client(): S3Client {
            // Configure your S3Client to use the Asymmetric Sigv4 (Sigv4a)
signing algorithm.
            val sigV4AScheme = SigV4AsymmetricAuthScheme(CrtAwsSigner)
            val s3 = S3Client.fromEnvironment {
                authSchemes = listOf(sigV4AScheme)
            }
            return s3
        }
```

Verwenden Sie den ARN des Multi-Region Access Points anstelle eines Bucket-Namens, um das Objekt abzurufen.

```
    suspend fun getObjectFromMrap(s3: S3Client, mrapArn: String, keyName:
String): String? {
        val request = GetObjectRequest {
            bucket = mrapArn // Use the ARN instead of the bucket name for object
operations.
            key = keyName
        }

        var stringObj: String? = null
        s3.getObject(request) { resp ->
            stringObj = resp.body?.decodeToString()
            if (stringObj != null) {
                println("Successfully read $keyName from $mrapArn")
            }
        }
        return stringObj
    }
```

- Weitere Informationen finden Sie im Entwicklerhandbuch zum AWS SDK für Kotlin.
- Weitere API-Informationen finden Sie unter GetObject in der API-AWS Referenz zum -SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen eines Objekts aus einem Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird veranschaulicht, wie Sie Daten aus einem Objekt in einem S3 Bucket lesen.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erste Schritte mit Buckets und Objekten
- Erste Schritte mit der Verschlüsselung

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Shows how to download an object from an Amazon S3 bucket to the
        /// local computer.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket where the object is
        /// currently stored.</param>
        /// <param name="objectName">The name of the object to download.</param>
        /// <param name="filePath">The path, including filename, where the
        /// downloaded object will be stored.</param>
        /// <returns>A boolean value indicating the success or failure of the
        /// download process.</returns>
        public static async Task<bool> DownloadObjectFromBucketAsync(
            IAmazonS3 client,
            string bucketName,
            string objectName,
            string filePath)
```

```
        {
            // Create a GetObject request
            var request = new GetObjectRequest
            {
                BucketName = bucketName,
                Key = objectName,
            };

            // Issue request and remember to dispose of the response
            using GetObjectResponse response = await
 client.GetObjectAsync(request);

            try
            {
                // Save object to local file
                await response.WriteResponseStreamToFileAsync($"{filePath}\
\{objectName}", true, CancellationToken.None);
                return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error saving {objectName}: {ex.Message}");
                return false;
            }
        }
```

- Weitere API-Informationen finden Sie unter GetObject in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#       $1 - The name of the bucket to download the object from.
#       $2 - The path and file name to store the downloaded bucket.
#       $3 - The key (name) of the object in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function download_object_from_bucket() {
  local bucket_name=$1
  local destination_file_name=$2
  local object_name=$3
  local response

  response=$(aws s3api get-object \
    --bucket "$bucket_name" \
    --key "$object_name" \
    "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}
```

- API-Details finden Sie unter GetObject in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::GetObject(const Aws::String &objectKey,
                           const Aws::String &fromBucket,
                           const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::GetObjectRequest request;
    request.SetBucket(fromBucket);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectOutcome outcome =
            client.GetObject(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: GetObject: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully retrieved '" << objectKey << "' from '"
                  << fromBucket << "'." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter GetObject in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Im folgenden Beispiel wird der `get-object` Befehl verwendet, um ein Objekt von Amazon S3 herunterzuladen:

```
aws s3api get-object --bucket text-content --key dir/my_images.tar.bz2
 my_images.tar.bz2
```

Beachten Sie, dass der Outfile-Parameter ohne Optionsnamen wie „--outfile“ angegeben wird. Der Name der Ausgabedatei muss der letzte Parameter im Befehl sein.

Das folgende Beispiel zeigt die Verwendung von `--range` zum Herunterladen eines bestimmten Bytebereichs von einem Objekt. Beachten Sie, dass den Bytebereichen das Präfix „bytes=“ vorangestellt werden muss:

```
aws s3api get-object --bucket text-content --key dir/my_data --range
 bytes=8888-9999 my_data_range
```

Weitere Informationen zum Abrufen von Objekten finden Sie unter Abrufen von Objekten im Amazon S3-Entwicklerhandbuch.

- API-Details finden Sie unter GetObject in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
```

```
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// DownloadFile gets an object from a bucket and stores it in a local file.
func (basics BucketBasics) DownloadFile(bucketName string, objectKey string,
 fileName string) error {
 result, err := basics.S3Client.GetObject(context.TODO(), &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't get object %v:%v. Here's why: %v\n", bucketName,
 objectKey, err)
  return err
 }
 defer result.Body.Close()
 file, err := os.Create(fileName)
 if err != nil {
  log.Printf("Couldn't create file %v. Here's why: %v\n", fileName, err)
  return err
 }
 defer file.Close()
 body, err := io.ReadAll(result.Body)
 if err != nil {
  log.Printf("Couldn't read object body from %v. Here's why: %v\n", objectKey,
 err)
 }
 _, err = file.Write(body)
 return err
}
```

- Weitere API-Informationen finden Sie unter GetObject in der APIAWS SDK for Go -Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Lesen Sie Daten als Byte-Array mit einem S3Client.

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetObjectData {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName> <path>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name.\s
```

```
                    path - The path where the file is written to.\s
                """";

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        String path = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getObjectBytes(s3, bucketName, keyName, path);
    }

    public static void getObjectBytes(S3Client s3, String bucketName, String
keyName, String path) {
        try {
            GetObjectRequest objectRequest = GetObjectRequest
                    .builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();

            ResponseBytes<GetObjectResponse> objectBytes =
s3.getObjectAsBytes(objectRequest);
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from an S3 object");
            os.close();

        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
    }
}
```

Verwenden Sie ein S3TransferManager, um ein Objekt in einem S3-Bucket in eine lokale Datei herunterzuladen. S3 Sehen Sie sich die vollständige Datei an und testen Sie sie.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedFileDownload;
import software.amazon.awssdk.transfer.s3.model.DownloadFileRequest;
import software.amazon.awssdk.transfer.s3.model.FileDownload;
import software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.UUID;

    public Long downloadFile(S3TransferManager transferManager, String
 bucketName,
                             String key, String downloadedFileWithPath) {
        DownloadFileRequest downloadFileRequest = DownloadFileRequest.builder()
            .getObjectRequest(b -> b.bucket(bucketName).key(key))
            .addTransferListener(LoggingTransferListener.create())
            .destination(Paths.get(downloadedFileWithPath))
            .build();

        FileDownload downloadFile =
 transferManager.downloadFile(downloadFileRequest);

        CompletedFileDownload downloadResult =
 downloadFile.completionFuture().join();
        logger.info("Content length [{}]",
 downloadResult.response().contentLength());
        return downloadResult.response().contentLength();
    }
```

Lesen Sie Tags, die zu einem Objekt gehören, mit einem S3Client.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingRequest;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Tag;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetObjectTags {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - A key name that represents the object.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();
```

```
        listTags(s3, bucketName, keyName);
        s3.close();
    }

    public static void listTags(S3Client s3, String bucketName, String keyName) {
        try {
            GetObjectTaggingRequest getTaggingRequest = GetObjectTaggingRequest
                    .builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();

            GetObjectTaggingResponse tags =
 s3.getObjectTagging(getTaggingRequest);
            List<Tag> tagSet = tags.tagSet();
            for (Tag tag : tagSet) {
                System.out.println(tag.key());
                System.out.println(tag.value());
            }

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Rufen Sie eine URL für ein Objekt mit einem S3Client ab.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetUrlRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.net.URL;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetObjectUrl {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.
                    keyName - A key name that represents the object.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getURL(s3, bucketName, keyName);
        s3.close();
    }

    public static void getURL(S3Client s3, String bucketName, String keyName) {
        try {
            GetUrlRequest request = GetUrlRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            URL url = s3.utilities().getUrl(request);
            System.out.println("The URL for  " + keyName + " is " + url);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

Rufen Sie ein Objekt mithilfe des S3Presigner-Client-Objekts mit einem S3Client ab.

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.time.Duration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import
 software.amazon.awssdk.services.s3.presigner.model.GetObjectPresignRequest;
import
 software.amazon.awssdk.services.s3.presigner.model.PresignedGetObjectRequest;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import software.amazon.awssdk.utils.IoUtils;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetObjectPresignedUrl {
    public static void main(String[] args) {
        final String USAGE = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - A key name that represents a text file.\s
                """;
```

```
        if (args.length != 2) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Presigner presigner = S3Presigner.builder()
                .region(region)
                .build();

        getPresignedUrl(presigner, bucketName, keyName);
        presigner.close();
    }

    public static void getPresignedUrl(S3Presigner presigner, String bucketName,
String keyName) {
        try {
            GetObjectRequest getObjectRequest = GetObjectRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            GetObjectPresignRequest getObjectPresignRequest =
GetObjectPresignRequest.builder()
                    .signatureDuration(Duration.ofMinutes(60))
                    .getObjectRequest(getObjectRequest)
                    .build();

            PresignedGetObjectRequest presignedGetObjectRequest =
presigner.presignGetObject(getObjectPresignRequest);
            String theUrl = presignedGetObjectRequest.url().toString();
            System.out.println("Presigned URL: " + theUrl);
            HttpURLConnection connection = (HttpURLConnection)
presignedGetObjectRequest.url().openConnection();
            presignedGetObjectRequest.httpRequest().headers().forEach((header,
values) -> {
                values.forEach(value -> {
                    connection.addRequestProperty(header, value);
                });
            });
```

```
            // Send any request payload that the service needs (not needed when
            // isBrowserExecutable is true).
            if (presignedGetObjectRequest.signedPayload().isPresent()) {
                connection.setDoOutput(true);

                try (InputStream signedPayload =
 presignedGetObjectRequest.signedPayload().get().asInputStream();
                        OutputStream httpOutputStream =
 connection.getOutputStream()) {
                    IoUtils.copy(signedPayload, httpOutputStream);
                }
            }

            // Download the result of executing the request.
            try (InputStream content = connection.getInputStream()) {
                System.out.println("Service returned response: ");
                IoUtils.copy(content, System.out);
            }

        } catch (S3Exception | IOException e) {
            e.getStackTrace();
        }
    }
}
```

Rufen Sie ein Objekt ab, indem Sie ein ResponseTransformer Objekt und S3Client verwenden.

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.core.sync.ResponseTransformer;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
```

```
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetDataResponseTransformer {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName> <path>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name.\s
                    path - The path where the file is written to.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        String path = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getObjectBytes(s3, bucketName, keyName, path);
        s3.close();
    }

    public static void getObjectBytes(S3Client s3, String bucketName, String
 keyName, String path) {
        try {
            GetObjectRequest objectRequest = GetObjectRequest
                    .builder()
                    .key(keyName)
                    .bucket(bucketName)
```

```
                    .build();

            ResponseBytes<GetObjectResponse> objectBytes =
 s3.getObject(objectRequest, ResponseTransformer.toBytes());
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from an S3 object");
            os.close();

        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter GetObject in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Laden Sie das Objekt herunter.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});
```

```
export const main = async () => {
  const command = new GetObjectCommand({
    Bucket: "test-bucket",
    Key: "hello-s3.txt",
  });

  try {
    const response = await client.send(command);
    // The Body object also has 'transformToByteArray' and 'transformToWebStream'
 methods.
    const str = await response.Body.transformToString();
    console.log(str);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter GetObject in der APIAWS SDK for JavaScript - Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun getObjectBytes(bucketName: String, keyName: String, path: String) {
    val request = GetObjectRequest {
        key = keyName
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.getObject(request) { resp ->
            val myFile = File(path)
```

```
            resp.body?.writeToFile(myFile)
            println("Successfully read $keyName from $bucketName")
        }
    }
}
```

- Weitere API-Informationen finden Sie unter GetObject in der AWS API-Referenz zum -SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie ein Objekt ab.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $file = $this->s3client->getObject([
                'Bucket' => $this->bucketName,
                'Key' => $fileName,
            ]);
            $body = $file->get('Body');
            $body->rewind();
            echo "Downloaded the file and it begins with: {$body->read(26)}.\n";
        } catch (Exception $exception) {
            echo "Failed to download $fileName from $this->bucketName with error:
 " . $exception->getMessage();
            exit("Please fix error with file downloading before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter GetObject in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def get(self):
        """
        Gets the object.

        :return: The object data in bytes.
        """
        try:
            body = self.object.get()["Body"].read()
            logger.info(
                "Got object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't get object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
```

```
            raise
        else:
            return body
```

- Weitere API-Informationen finden Sie unter GetObject in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie ein Objekt ab.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectGetWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object directly to a file.
  #
  # @param target_path [String] The path to the file where the object is
 downloaded.
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
 successful; otherwise nil.
  def get_object(target_path)
    @object.get(response_target: target_path)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
```

```
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"
  target_path = "my-object-as-file.txt"

  wrapper = ObjectGetWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  obj_data = wrapper.get_object(target_path)
  return unless obj_data

  puts "Object #{object_key} (#{obj_data.content_length} bytes} downloaded to
 #{target_path}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Rufen Sie ein Objekt ab und melden Sie seinen serverseitigen Verschlüsselungsstatus.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectGetEncryptionWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object into memory.
  #
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
 successful; otherwise nil.
  def get_object
    @object.get
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end
```

```
# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectGetEncryptionWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  obj_data = wrapper.get_object
  return unless obj_data

  encryption = obj_data.server_side_encryption.nil? ? "no" :
 obj_data.server_side_encryption
  puts "Object #{object_key} uses #{encryption} encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter GetObject in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn get_object(client: Client, opt: Opt) -> Result<usize, anyhow::Error> {
    trace!("bucket:      {}", opt.bucket);
    trace!("object:      {}", opt.object);
    trace!("destination: {}", opt.destination.display());

    let mut file = File::create(opt.destination.clone())?;

    let mut object = client
        .get_object()
```

```
        .bucket(opt.bucket)
        .key(opt.object)
        .send()
        .await?;

    let mut byte_count = 0_usize;
    while let Some(bytes) = object.body.try_next().await? {
        let bytes_len = bytes.len();
        file.write_all(&bytes)?;
        trace!("Intermediate write of {bytes_len}");
        byte_count += bytes_len;
    }

    Ok(byte_count)
}
```

- Weitere API-Informationen finden Sie unter GetObject in der AWS API-Referenz zum -SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    TRY.
        oo_result = lo_s3->getobject(           " oo_result is returned for
testing purposes. "
                    iv_bucket = iv_bucket_name
                    iv_key = iv_object_key
                ).
        DATA(lv_object_data) = oo_result->get_body( ).
        MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
      CATCH /aws1/cx_s3_nosuchkey.
```

```
        MESSAGE 'Object key does not exist.' TYPE 'E'.
    ENDTRY.
```

- Weitere API-Informationen finden Sie unter GetObject in der AWS API-Referenz zum -SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Ein Objekt von einem Bucket in eine lokale Datei herunterladen

```
    public func downloadFile(bucket: String, key: String, to: String) async
throws {
        let fileUrl = URL(fileURLWithPath: to).appendingPathComponent(key)

        let input = GetObjectInput(
            bucket: bucket,
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the data stream object. Return immediately if there isn't one.
        guard let body = output.body,
              let data = try await body.readData() else {
            return
        }
```

```
        try data.write(to: fileUrl)
    }
```

Ein Objekt in ein Swift-Data-Objekt lesen

```
    public func readFile(bucket: String, key: String) async throws -> Data {
        let input = GetObjectInput(
            bucket: bucket,
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the stream and return its contents in a `Data` object. If
        // there is no stream, return an empty `Data` object instead.
        guard let body = output.body,
              let data = try await body.readData() else {
            return "".data(using: .utf8)!
        }

        return data
    }
```

- Weitere API-Informationen finden Sie unter GetObject in der AWS API-Referenz zum -SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen eines Objekts aus einem Amazon S3-Bucket mithilfe eines AWS -SDK unter Angabe eines If-Modified-Since-Headers

Das folgende Codebeispiel zeigt, wie Sie Daten aus einem Objekt in einem S3 Bucket lesen, jedoch nur, wenn dieser Bucket seit dem letzten Abruf nicht geändert wurde.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erste Schritte mit Buckets und Objekten

- Erste Schritte mit der Verschlüsselung

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
use aws_sdk_s3::{
    error::SdkError,
    operation::head_object::HeadObjectError,
    primitives::{ByteStream, DateTime, DateTimeFormat},
    Client, Error,
};
use tracing::{error, warn};

const KEY: &str = "key";
const BODY: &str = "Hello, world!";

/// Demonstrate how `if-modified-since` reports that matching objects haven't
/// changed.
///
/// # Steps
/// - Create a bucket.
/// - Put an object in the bucket.
/// - Get the bucket headers.
/// - Get the bucket headers again but only if modified.
/// - Delete the bucket.
#[tokio::main]
async fn main() -> Result<(), Error> {
    tracing_subscriber::fmt::init();

    // Get a new UUID to use when creating a unique bucket name.
    let uuid = uuid::Uuid::new_v4();

    // Load the AWS configuration from the environment.
    let client = Client::new(&aws_config::load_from_env().await);
```

```
    // Generate a unique bucket name using the previously generated UUID.
    // Then create a new bucket with that name.
    let bucket_name = format!("if-modified-since-{uuid}");
    client
        .create_bucket()
        .bucket(bucket_name.clone())
        .send()
        .await?;

    // Create a new object in the bucket whose name is `KEY` and whose
    // contents are `BODY`.
    let put_object_output = client
        .put_object()
        .bucket(bucket_name.as_str())
        .key(KEY)
        .body(ByteStream::from_static(BODY.as_bytes()))
        .send()
        .await;

    // If the `PutObject` succeeded, get the eTag string from it. Otherwise,
    // report an error and return an empty string.
    let e_tag_1 = match put_object_output {
        Ok(put_object) => put_object.e_tag.unwrap(),
        Err(err) => {
            error!("{err:?}");
            String::new()
        }
    };

    // Request the object's headers.
    let head_object_output = client
        .head_object()
        .bucket(bucket_name.as_str())
        .key(KEY)
        .send()
        .await;

    // If the `HeadObject` request succeeded, create a tuple containing the
    // values of the headers `last-modified` and `etag`. If the request
    // failed, return the error in a tuple instead.
    let (last_modified, e_tag_2) = match head_object_output {
        Ok(head_object) => (
            Ok(head_object.last_modified().cloned().unwrap()),
            head_object.e_tag.unwrap(),
```

```
        ),
        Err(err) => (Err(err), String::new()),
    };

    warn!("last modified: {last_modified:?}");
    assert_eq!(
        e_tag_1, e_tag_2,
        "PutObject and first GetObject had differing eTags"
    );

    println!("First value of last_modified: {last_modified:?}");
    println!("First tag: {}\n", e_tag_1);

    // Send a second `HeadObject` request. This time, the `if_modified_since`
    // option is specified, giving the `last_modified` value returned by the
    // first call to `HeadObject`.
    //
    // Since the object hasn't been changed, and there are no other objects in
    // the bucket, there should be no matching objects.

    let head_object_output = client
        .head_object()
        .bucket(bucket_name.as_str())
        .key(KEY)
        .if_modified_since(last_modified.unwrap())
        .send()
        .await;

    // If the `HeadObject` request succeeded, the result is a typle containing
    // the `last_modified` and `e_tag_1` properties. This is _not_ the expected
    // result.
    //
    // The _expected_ result of the second call to `HeadObject` is an
    // `SdkError::ServiceError` containing the HTTP error response. If that's
    // the case and the HTTP status is 304 (not modified), the output is a
    // tuple containing the values of the HTTP `last-modified` and `etag`
    // headers.
    //
    // If any other HTTP error occurred, the error is returned as an
    // `SdkError::ServiceError`.

    let (last_modified, e_tag_2): (Result<DateTime, SdkError<HeadObjectError>>,
String) =
        match head_object_output {
```

```
            Ok(head_object) => (
                Ok(head_object.last_modified().cloned().unwrap()),
                head_object.e_tag.unwrap(),
            ),
            Err(err) => match err {
                SdkError::ServiceError(err) => {
                    // Get the raw HTTP response. If its status is 304, the
                    // object has not changed. This is the expected code path.
                    let http = err.raw();
                    match http.status().as_u16() {
                        // If the HTTP status is 304: Not Modified, return a
                        // tuple containing the values of the HTTP
                        // `last-modified` and `etag` headers.
                        304 => (
                            Ok(DateTime::from_str(
                                http.headers().get("last-modified").unwrap(),
                                DateTimeFormat::HttpDate,
                            )
                            .unwrap()),
                            http.headers().get("etag").map(|t|
t.into()).unwrap(),
                        ),
                        // Any other HTTP status code is returned as an
                        // `SdkError::ServiceError`.
                        _ => (Err(SdkError::ServiceError(err)), String::new()),
                    }
                }
                // Any other kind of error is returned in a tuple containing the
                // error and an empty string.
                _ => (Err(err), String::new()),
            },
        };

    warn!("last modified: {last_modified:?}");
    assert_eq!(
        e_tag_1, e_tag_2,
        "PutObject and second HeadObject had different eTags"
    );

    println!("Second value of last modified: {last_modified:?}");
    println!("Second tag: {}", e_tag_2);

    // Clean up by deleting the object and the bucket.
    client
```

```
        .delete_object()
        .bucket(bucket_name.as_str())
        .key(KEY)
        .send()
        .await?;

    client
        .delete_bucket()
        .bucket(bucket_name.as_str())
        .send()
        .await?;

    Ok(())
}
```

- Weitere API-Informationen finden Sie unter GetObject in der API-AWS Referenz zum -SDK für Rust .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der ACL eines Amazon S3-Buckets mithilfe eines - AWS SDK

Die folgenden Codebeispiele veranschaulichen, wie Sie die Zugriffssteuerungsliste (Access Control List, ACL) eines S3 Buckets abrufen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Zugriffssteuerungslisten (ACL)

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Get the access control list (ACL) for the new bucket.
        /// </summary>
        /// <param name="client">The initialized client object used to get the
        /// access control list (ACL) of the bucket.</param>
        /// <param name="newBucketName">The name of the newly created bucket.</
param>
        /// <returns>An S3AccessControlList.</returns>
        public static async Task<S3AccessControlList>
 GetACLForBucketAsync(IAmazonS3 client, string newBucketName)
        {
            // Retrieve bucket ACL to show that the ACL was properly applied to
            // the new bucket.
            GetACLResponse getACLResponse = await client.GetACLAsync(new
 GetACLRequest
            {
                BucketName = newBucketName,
            });

            return getACLResponse.AccessControlList;
        }
```

- Weitere API-Informationen finden Sie unter GetBucketAcl in der APIAWS SDK for .NET - Referenz für .

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::GetBucketAcl(const Aws::String &bucketName,
                              const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::GetBucketAclRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketAclOutcome outcome =
            s3_client.GetBucketAcl(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: GetBucketAcl: "
                  << err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        Aws::Vector<Aws::S3::Model::Grant> grants =
                outcome.GetResult().GetGrants();

        for (auto it = grants.begin(); it != grants.end(); it++) {
            Aws::S3::Model::Grant grant = *it;
            Aws::S3::Model::Grantee grantee = grant.GetGrantee();

            std::cout << "For bucket " << bucketName << ": "
                      << std::endl << std::endl;

            if (grantee.TypeHasBeenSet()) {
                std::cout << "Type:          "
                          << GetGranteeTypeString(grantee.GetType()) <<
 std::endl;
```

```
            }

            if (grantee.DisplayNameHasBeenSet()) {
                std::cout << "Display name:  "
                          << grantee.GetDisplayName() << std::endl;
            }

            if (grantee.EmailAddressHasBeenSet()) {
                std::cout << "Email address: "
                          << grantee.GetEmailAddress() << std::endl;
            }

            if (grantee.IDHasBeenSet()) {
                std::cout << "ID:            "
                          << grantee.GetID() << std::endl;
            }

            if (grantee.URIHasBeenSet()) {
                std::cout << "URI:           "
                          << grantee.GetURI() << std::endl;
            }

            std::cout << "Permission:    " <<
                      GetPermissionString(grant.GetPermission()) <<
                      std::endl << std::endl;
        }
    }

    return outcome.IsSuccess();
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
/*!
 \sa GetGranteeTypeString()
 \param type Type enumeration.
*/

Aws::String GetGranteeTypeString(const Aws::S3::Model::Type &type) {
    switch (type) {
        case Aws::S3::Model::Type::AmazonCustomerByEmail:
            return "Email address of an AWS account";
        case Aws::S3::Model::Type::CanonicalUser:
            return "Canonical user ID of an AWS account";
```

```
        case Aws::S3::Model::Type::Group:
            return "Predefined Amazon S3 group";
        case Aws::S3::Model::Type::NOT_SET:
            return "Not set";
        default:
            return "Type unknown";
    }
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
/*!
 \sa GetPermissionString()
 \param permission Permission enumeration.
*/

Aws::String GetPermissionString(const Aws::S3::Model::Permission &permission) {
    switch (permission) {
        case Aws::S3::Model::Permission::FULL_CONTROL:
            return "Can list objects in this bucket, create/overwrite/delete "
                   "objects in this bucket, and read/write this "
                   "bucket's permissions";
        case Aws::S3::Model::Permission::NOT_SET:
            return "Permission not set";
        case Aws::S3::Model::Permission::READ:
            return "Can list objects in this bucket";
        case Aws::S3::Model::Permission::READ_ACP:
            return "Can read this bucket's permissions";
        case Aws::S3::Model::Permission::WRITE:
            return "Can create, overwrite, and delete objects in this bucket";
        case Aws::S3::Model::Permission::WRITE_ACP:
            return "Can write this bucket's permissions";
        default:
            return "Permission unknown";
    }

    return "Permission unknown";
}
```

- Weitere API-Informationen finden Sie unter GetBucketAcl in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl ruft die Zugriffskontrollliste für einen Bucket mit dem Namen abmy-bucket:

```
aws s3api get-bucket-acl --bucket my-bucket
```

Ausgabe:

```
{
    "Owner": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
            },
            "Permission": "FULL_CONTROL"
        }
    ]
}
```

- API-Details finden Sie unter GetBucketAcl in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.s3.model.S3Exception;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectAclRequest;
import software.amazon.awssdk.services.s3.model.GetObjectAclResponse;
import software.amazon.awssdk.services.s3.model.Grant;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetAcl {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <bucketName> <objectKey>

                Where:
                  bucketName - The Amazon S3 bucket to get the access control
 list (ACL) for.
                  objectKey - The object to get the ACL for.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String objectKey = args[1];
        System.out.println("Retrieving ACL for object: " + objectKey);
        System.out.println("in bucket: " + bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();
```

```
        getBucketACL(s3, objectKey, bucketName);
        s3.close();
        System.out.println("Done!");
    }

    public static String getBucketACL(S3Client s3, String objectKey, String
 bucketName) {
        try {
            GetObjectAclRequest aclReq = GetObjectAclRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .build();

            GetObjectAclResponse aclRes = s3.getObjectAcl(aclReq);
            List<Grant> grants = aclRes.grants();
            String grantee = "";
            for (Grant grant : grants) {
                System.out.format("  %s: %s\n", grant.grantee().id(),
 grant.permission());
                grantee = grant.grantee().id();
            }

            return grantee;
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return "";
    }
}
```

- Weitere API-Informationen finden Sie unter GetBucketAcl in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die ACL-Berechtigungen ab.

```
import { GetBucketAclCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new GetBucketAclCommand({
    Bucket: "test-bucket",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter GetBucketAcl in der APIAWS SDK for JavaScript -Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def get_acl(self):
        """
        Get the ACL of the bucket.

        :return: The ACL of the bucket.
        """
        try:
            acl = self.bucket.Acl()
            logger.info(
                "Got ACL for bucket %s. Owner is %s.", self.bucket.name,
 acl.owner
            )
        except ClientError:
            logger.exception("Couldn't get ACL for bucket %s.", self.bucket.name)
            raise
        else:
            return acl
```

- Weitere API-Informationen finden Sie unter GetBucketAcl in der AWS API-Referenz zum - SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der ACL eines Amazon S3-Objekts mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Zugriffssteuerungsliste (Access Control List, ACL) eines S3-Objekts abrufen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Zugriffssteuerungslisten (ACL)

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::GetObjectAcl(const Aws::String &bucketName,
                              const Aws::String &objectKey,
                              const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::GetObjectAclRequest request;
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectAclOutcome outcome =
            s3_client.GetObjectAcl(request);
```

```
   if (!outcome.IsSuccess()) {
       const Aws::S3::S3Error &err = outcome.GetError();
       std::cerr << "Error: GetObjectAcl: "
                 << err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
   }
   else {
       Aws::Vector<Aws::S3::Model::Grant> grants =
               outcome.GetResult().GetGrants();

       for (auto it = grants.begin(); it != grants.end(); it++) {
           std::cout << "For object " << objectKey << ": "
                     << std::endl << std::endl;

           Aws::S3::Model::Grant grant = *it;
           Aws::S3::Model::Grantee grantee = grant.GetGrantee();

           if (grantee.TypeHasBeenSet()) {
               std::cout << "Type:          "
                         << GetGranteeTypeString(grantee.GetType()) <<
std::endl;
           }

           if (grantee.DisplayNameHasBeenSet()) {
               std::cout << "Display name:  "
                         << grantee.GetDisplayName() << std::endl;
           }

           if (grantee.EmailAddressHasBeenSet()) {
               std::cout << "Email address: "
                         << grantee.GetEmailAddress() << std::endl;
           }

           if (grantee.IDHasBeenSet()) {
               std::cout << "ID:            "
                         << grantee.GetID() << std::endl;
           }

           if (grantee.URIHasBeenSet()) {
               std::cout << "URI:           "
                         << grantee.GetURI() << std::endl;
           }

           std::cout << "Permission:    " <<
```

```
                        GetPermissionString(grant.GetPermission()) <<
                        std::endl << std::endl;
        }
    }

    return outcome.IsSuccess();
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
/*!
 \fn GetGranteeTypeString()
 \param type Type enumeration.
*/

Aws::String GetGranteeTypeString(const Aws::S3::Model::Type &type) {
    switch (type) {
        case Aws::S3::Model::Type::AmazonCustomerByEmail:
            return "Email address of an AWS account";
        case Aws::S3::Model::Type::CanonicalUser:
            return "Canonical user ID of an AWS account";
        case Aws::S3::Model::Type::Group:
            return "Predefined Amazon S3 group";
        case Aws::S3::Model::Type::NOT_SET:
            return "Not set";
        default:
            return "Type unknown";
    }
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
/*!
 \fn GetPermissionString()
 \param permission Permission enumeration.
*/

Aws::String GetPermissionString(const Aws::S3::Model::Permission &permission) {
    switch (permission) {
        case Aws::S3::Model::Permission::FULL_CONTROL:
            return "Can read this object's data and its metadata, "
                   "and read/write this object's permissions";
        case Aws::S3::Model::Permission::NOT_SET:
            return "Permission not set";
```

```
        case Aws::S3::Model::Permission::READ:
            return "Can read this object's data and its metadata";
        case Aws::S3::Model::Permission::READ_ACP:
            return "Can read this object's permissions";
            // case Aws::S3::Model::Permission::WRITE // Not applicable.
        case Aws::S3::Model::Permission::WRITE_ACP:
            return "Can write this object's permissions";
        default:
            return "Permission unknown";
    }
}
```

- Weitere API-Informationen finden Sie unter GetObjectAcl in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl ruft die Zugriffskontrollliste für ein Objekt in einem Bucket mit dem Namen abmy-bucket:

```
aws s3api get-object-acl --bucket my-bucket --key index.html
```

Ausgabe:

```
{
    "Owner": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
            },
            "Permission": "FULL_CONTROL"
        },
```

```
        {
            "Grantee": {
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

- API-Details finden Sie unter GetObjectAcl in der AWS CLI -Befehlsreferenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun getBucketACL(objectKey: String, bucketName: String) {
    val request = GetObjectAclRequest {
        bucket = bucketName
        key = objectKey
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        val response = s3.getObjectAcl(request)
        response.grants?.forEach { grant ->
            println("Grant permission is ${grant.permission}")
        }
    }
}
```

- Weitere API-Informationen finden Sie unter GetObjectAcl in der AWS API-Referenz zum -SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def get_acl(self):
        """
        Gets the ACL of the object.

        :return: The ACL of the object.
        """
        try:
            acl = self.object.Acl()
            logger.info(
                "Got ACL for object %s owned by %s.",
                self.object.key,
                acl.owner["DisplayName"],
            )
        except ClientError:
            logger.exception("Couldn't get ACL for object %s.", self.object.key)
            raise
        else:
            return acl
```

- Weitere API-Informationen finden Sie unter GetObjectAcl in der AWS API-Referenz zum - SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der Region, in der sich der Amazon S3-Bucket befindet, mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie den Standort der Region für einen S3-Bucket abrufen.

CLI

AWS CLI

Der folgende Befehl ruft die Standortbeschränkung für einen Bucket mit dem Namen ab`my-bucket`, wenn eine Einschränkung vorhanden ist:

```
aws s3api get-bucket-location --bucket my-bucket
```

Ausgabe:

```
{
    "LocationConstraint": "us-west-2"
}
```

- API-Details finden Sie unter GetBucketLocation in der AWS CLI -Befehlsreferenz.

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn show_buckets(strict: bool, client: &Client, region: &str) -> Result<(),
 Error> {
    let resp = client.list_buckets().send().await?;
    let buckets = resp.buckets();
    let num_buckets = buckets.len();

    let mut in_region = 0;

    for bucket in buckets {
        if strict {
            let r = client
                .get_bucket_location()
                .bucket(bucket.name().unwrap_or_default())
                .send()
                .await?;

            if r.location_constraint().unwrap().as_ref() == region {
                println!("{}", bucket.name().unwrap_or_default());
                in_region += 1;
            }
        } else {
            println!("{}", bucket.name().unwrap_or_default());
        }
    }

    println!();
    if strict {
        println!(
            "Found {} buckets in the {} region out of a total of {} buckets.",
            in_region, region, num_buckets
        );
    } else {
```

```
        println!("Found {} buckets in all regions.", num_buckets);
    }

    Ok(())
}
```

- Weitere API-Informationen finden Sie unter GetBucketLocation in der APIAWS -Referenz zum -SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie die Konfiguration für die gesetzliche Aufbewahrungsfrist eines S3-Buckets abrufen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Sperren von Amazon S3-Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Get the legal hold details for an S3 object.
    /// </summary>
```

```
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object legal hold details.</returns>
    public async Task<ObjectLockLegalHold> GetObjectLegalHold(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectLegalHoldAsync(request);
            Console.WriteLine($"\tObject legal hold for {objectKey} in
{bucketName}: " +
                              $"\n\tStatus: {response.LegalHold.Status}");
            return response.LegalHold;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch legal hold: '{ex.Message}'");
            return new ObjectLockLegalHold();
        }
    }
```

- Weitere API-Informationen finden Sie unter GetObjectLegalHold in der APIAWS SDK for .NET -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der Lebenszykluskonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Lebenszyklus-Konfiguration eines S3-Buckets abrufen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Returns a configuration object for the supplied bucket name.
        /// </summary>
        /// <param name="client">The S3 client object used to call
        /// the GetLifecycleConfigurationAsync method.</param>
        /// <param name="bucketName">The name of the S3 bucket for which a
        /// configuration will be created.</param>
        /// <returns>Returns a new LifecycleConfiguration object.</returns>
        public static async Task<LifecycleConfiguration>
RetrieveLifecycleConfigAsync(IAmazonS3 client, string bucketName)
        {
            var request = new GetLifecycleConfigurationRequest()
            {
                BucketName = bucketName,
            };
            var response = await client.GetLifecycleConfigurationAsync(request);
            var configuration = response.Configuration;
            return configuration;
        }
```

- Weitere API-Informationen finden Sie unter GetBucketLifecycleConfiguration in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

Der folgende Befehl ruft die Lebenszykluskonfiguration für einen Bucket mit dem Namen ab`my-bucket`:

```
aws s3api get-bucket-lifecycle-configuration --bucket my-bucket
```

Ausgabe:

```
{
    "Rules": [
        {
            "ID": "Move rotated logs to Glacier",
            "Prefix": "rotated/",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Date": "2015-11-10T00:00:00.000Z",
                    "StorageClass": "GLACIER"
                }
            ]
        },
        {
            "Status": "Enabled",
            "Prefix": "",
            "NoncurrentVersionTransitions": [
                {
                    "NoncurrentDays": 0,
                    "StorageClass": "GLACIER"
                }
            ],
            "ID": "Move old versions to Glacier"
        }
    ]
}
```

- API-Details finden Sie unter GetBucketLifecycleConfiguration in der AWS CLI - Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def get_lifecycle_configuration(self):
        """
        Get the lifecycle configuration of the bucket.

        :return: The lifecycle rules of the specified bucket.
        """
        try:
            config = self.bucket.LifecycleConfiguration()
            logger.info(
                "Got lifecycle rules %s for bucket '%s'.",
                config.rules,
                self.bucket.name,
            )
        except:
            logger.exception(
                "Couldn't get lifecycle rules for bucket '%s'.", self.bucket.name
            )
            raise
        else:
```

```
            return config.rules
```

- Weitere API-Informationen finden Sie unter GetBucketLifecycleConfiguration in der AWS API-Referenz zum -SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Objektsperrenkonfiguration eines S3-Buckets abrufen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Sperren von Amazon S3-Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Get the object lock configuration details for an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket to get details.</param>
    /// <returns>The bucket's object lock configuration details.</returns>
    public async Task<ObjectLockConfiguration>
GetBucketObjectLockConfiguration(string bucketName)
```

```
    {
        try
        {
            var request = new GetObjectLockConfigurationRequest()
            {
                BucketName = bucketName
            };

            var response = await
_amazonS3.GetObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tBucket object lock config for {bucketName} in
{bucketName}: " +
                              $"\n\tEnabled:
{response.ObjectLockConfiguration.ObjectLockEnabled}" +
                              $"\n\tRule:
{response.ObjectLockConfiguration.Rule?.DefaultRetention}");

            return response.ObjectLockConfiguration;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch object lock config:
'{ex.Message}'");
            return new ObjectLockConfiguration();
        }
    }
```

- Weitere API-Informationen finden Sie unter GetObjectLockConfiguration in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

So rufen Sie eine Objektsperrenkonfiguration für einen Bucket ab

Im folgenden `get-object-lock-configuration` Beispiel wird die Objektsperrkonfiguration für den angegebenen Bucket abgerufen.

```
aws s3api get-object-lock-configuration \
    --bucket my-bucket-with-object-lock
```

Ausgabe:

```
{
    "ObjectLockConfiguration": {
        "ObjectLockEnabled": "Enabled",
        "Rule": {
            "DefaultRetention": {
                "Mode": "COMPLIANCE",
                "Days": 50
            }
        }
    }
}
```

- API-Details finden Sie unter GetObjectLockConfiguration in der AWS CLI -Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der Richtlinie für einen Amazon S3-Bucket mithilfe eines - AWS SDK

Die folgenden Codebeispielen veranschaulichen, wie Sie die Richtlinie für einen S3 Bucket abrufen.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::GetBucketPolicy(const Aws::String &bucketName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);
```

```
    Aws::S3::Model::GetBucketPolicyRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketPolicyOutcome outcome =
            s3_client.GetBucketPolicy(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: GetBucketPolicy: "
                  << err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        Aws::StringStream policy_stream;
        Aws::String line;

        outcome.GetResult().GetPolicy() >> line;
        policy_stream << line;

        std::cout << "Retrieve the policy for bucket '" << bucketName << "':\n\n"
 <<
                  policy_stream.str() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter GetBucketPolicy in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl ruft die Bucket-Richtlinie für einen Bucket mit dem Namen ab`my-bucket`:

```
aws s3api get-bucket-policy --bucket my-bucket
```

Ausgabe:

```
{
```

```
    "Policy": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":\"\",\"Effect
\":\"Allow\",\"Principal\":\"*\",\"Action\":\"s3:GetObject\",\"Resource\":
\"arn:aws:s3:::my-bucket/*\"},{\"Sid\":\"\",\"Effect\":\"Deny\",\"Principal\":
\"*\",\"Action\":\"s3:GetObject\",\"Resource\":\"arn:aws:s3:::my-bucket/secret/*
\"}]}"
}
```

Bucket-policyThe folgende Beispiel zeigt, wie Sie eine Amazon S3-Bucket-Richtlinie herunterladen, Änderungen an der Datei vornehmen und dann verwenden können, `put-bucket-policy` um die geänderte Bucket-Richtlinie anzuwenden. Um die Bucket-Richtlinie in eine Datei herunterzuladen, können Sie Folgendes ausführen:

```
aws s3api get-bucket-policy --bucket mybucket --query Policy --output text >
 policy.json
```

Anschließend können Sie die `policy.json` Datei nach Bedarf ändern. Schließlich können Sie diese geänderte Richtlinie wieder auf den S3-Bucket anwenden, indem Sie Folgendes ausführen:

`policy.json` -Datei nach Bedarf. Schließlich können Sie diese geänderte Richtlinie wieder auf den S3-Bucket anwenden, indem Sie Folgendes ausführen:

-Datei nach Bedarf. Schließlich können Sie diese geänderte Richtlinie wieder auf den S3-Bucket anwenden, indem Sie Folgendes ausführen:

```
aws s3api put-bucket-policy --bucket mybucket --policy file://policy.json
```

- API-Details finden Sie unter GetBucketPolicy in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetBucketPolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>

                Where:
                    bucketName - The Amazon S3 bucket to get the policy from.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Getting policy for bucket: \"%s\"\n\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        String polText = getPolicy(s3, bucketName);
        System.out.println("Policy Text: " + polText);
        s3.close();
    }
```

```
    public static String getPolicy(S3Client s3, String bucketName) {
        String policyText;
        System.out.format("Getting policy for bucket: \"%s\"\n\n", bucketName);
        GetBucketPolicyRequest policyReq = GetBucketPolicyRequest.builder()
                .bucket(bucketName)
                .build();

        try {
            GetBucketPolicyResponse policyRes = s3.getBucketPolicy(policyReq);
            policyText = policyRes.policy();
            return policyText;

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return "";
    }
}
```

- Weitere API-Informationen finden Sie unter GetBucketPolicy in der APIAWS SDK for Java 2.x -Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die Bucket-Richtlinie ab.

```
import { GetBucketPolicyCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});
```

```
export const main = async () => {
  const command = new GetBucketPolicyCommand({
    Bucket: "test-bucket",
  });

  try {
    const { Policy } = await client.send(command);
    console.log(JSON.parse(Policy));
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter GetBucketPolicy in der APIAWS SDK for JavaScript -Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun getPolicy(bucketName: String): String? {
    println("Getting policy for bucket $bucketName")

    val request = GetBucketPolicyRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        val policyRes = s3.getBucketPolicy(request)
        return policyRes.policy
    }
}
```

- Weitere API-Informationen finden Sie unter GetBucketPolicy in der API-AWS Referenz zum -SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def get_policy(self):
        """
        Get the security policy of the bucket.

        :return: The security policy of the specified bucket, in JSON format.
        """
        try:
            policy = self.bucket.Policy()
            logger.info(
                "Got policy %s for bucket '%s'.", policy.policy, self.bucket.name
            )
        except ClientError:
```

```
            logger.exception("Couldn't get policy for bucket '%s'.",
 self.bucket.name)
            raise
        else:
            return json.loads(policy.policy)
```

- Weitere API-Informationen finden Sie unter GetBucketPolicy in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
 configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Gets the policy of a bucket.
  #
  # @return [Aws::S3::GetBucketPolicyOutput, nil] The current bucket policy.
  def get_policy
    policy = @bucket_policy.data.policy
    policy.respond_to?(:read) ? policy.read : policy
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get the policy for #{@bucket_policy.bucket.name}. Here's why:
 #{e.message}"
    nil
```

```
  end

end
```

- Weitere API-Informationen finden Sie unter GetBucketPolicy in der APIAWS SDK for Ruby - Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der Aufbewahrungskonfiguration eines Amazon S3-Objekts mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Aufbewahrungskonfiguration eines S3-Objekts abrufen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Sperren von Amazon S3-Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/// <summary>
/// Get the retention period for an S3 object.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The object key.</param>
/// <returns>The object retention details.</returns>
```

```
    public async Task<ObjectLockRetention> GetObjectRetention(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectRetentionRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectRetentionAsync(request);
            Console.WriteLine($"\tObject retention for {objectKey} in
{bucketName}: " +
                              $"\n\t{response.Retention.Mode} until
{response.Retention.RetainUntilDate:d}.");
            return response.Retention;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch object lock retention:
'{ex.Message}'");
            return new ObjectLockRetention();
        }
    }
```

- Weitere API-Informationen finden Sie unter GetObjectRetention in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

So rufen Sie die Objektaufbewahrungskonfiguration für ein Objekt ab

Im folgenden `get-object-retention` Beispiel wird die Objektaufbewahrungskonfiguration für das angegebene Objekt abgerufen.

```
aws s3api get-object-retention \
    --bucket my-bucket-with-object-lock \
    --key doc1.rtf
```

Ausgabe:

```
{
    "Retention": {
        "Mode": "GOVERNANCE",
        "RetainUntilDate": "2025-01-01T00:00:00.000Z"
    }
}
```

- API-Details finden Sie unter GetObjectRetention in der AWS CLI -Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen der Website-Konfiguration für einen Amazon S3-Bucket mithilfe eines - AWS SDK

Das folgende Beispiel veranschaulicht, wie Sie die Website-Konfiguration für einen S3 Bucket abrufen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
                // Get the website configuration.
                GetBucketWebsiteRequest getRequest = new
GetBucketWebsiteRequest()
                {
                    BucketName = bucketName,
                };
                GetBucketWebsiteResponse getResponse = await
client.GetBucketWebsiteAsync(getRequest);
```

```
                Console.WriteLine($"Index document:
{getResponse.WebsiteConfiguration.IndexDocumentSuffix}");
                Console.WriteLine($"Error document:
{getResponse.WebsiteConfiguration.ErrorDocument}");
```

- Weitere API-Informationen finden Sie unter GetBucketWebsite in der APIAWS SDK for .NET -Referenz für .

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::GetWebsiteConfig(const Aws::String &bucketName,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::GetBucketWebsiteRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketWebsiteOutcome outcome =
            s3_client.GetBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();

        std::cerr << "Error: GetBucketWebsite: "
                  << err.GetMessage() << std::endl;
    }
    else {
        Aws::S3::Model::GetBucketWebsiteResult websiteResult =
 outcome.GetResult();
```

```
        std::cout << "Success: GetBucketWebsite: "
                  << std::endl << std::endl
                  << "For bucket '" << bucketName << "':"
                  << std::endl
                  << "Index page : "
                  << websiteResult.GetIndexDocument().GetSuffix()
                  << std::endl
                  << "Error page: "
                  << websiteResult.GetErrorDocument().GetKey()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter GetBucketWebsite in der APIAWS SDK for C++ -Referenz für .

CLI

AWS CLI

Der folgende Befehl ruft die statische Website-Konfiguration für einen Bucket mit dem Namen ab`my-bucket`:

```
aws s3api get-bucket-website --bucket my-bucket
```

Ausgabe:

```
{
    "IndexDocument": {
        "Suffix": "index.html"
    },
    "ErrorDocument": {
        "Key": "error.html"
    }
}
```

- API-Details finden Sie unter GetBucketWebsite in der AWS CLI -Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die Website-Konfiguration ab.

```
import { GetBucketWebsiteCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new GetBucketWebsiteCommand({
    Bucket: "test-bucket",
  });

  try {
    const { ErrorDocument, IndexDocument } = await client.send(command);
    console.log(
      `Your bucket is set up to host a website. It has an error document:`,
      `${ErrorDocument.Key}, and an index document: ${IndexDocument.Suffix}.`,
    );
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere API-Informationen finden Sie unter GetBucketWebsite in der APIAWS SDK for JavaScript -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Auflisten von Amazon S3-Buckets mithilfe eines - AWS SDK

Die folgenden Code-Beispiele zeigen, wie Sie S3 Buckets auflisten.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
namespace ListBucketsExample
{
    using System;
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example uses the AWS SDK for .NET to list the Amazon Simple Storage
    /// Service (Amazon S3) buckets belonging to the default account.
    /// </summary>
    public class ListBuckets
    {
        private static IAmazonS3 _s3Client;

        /// <summary>
        /// Get a list of the buckets owned by the default user.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <returns>The response from the ListingBuckets call that contains a
        /// list of the buckets owned by the default user.</returns>
        public static async Task<ListBucketsResponse> GetBuckets(IAmazonS3
 client)
        {
            return await client.ListBucketsAsync();
        }
```

```
        /// <summary>
        /// This method lists the name and creation date for the buckets in
        /// the passed List of S3 buckets.
        /// </summary>
        /// <param name="bucketList">A List of S3 bucket objects.</param>
        public static void DisplayBucketList(List<S3Bucket> bucketList)
        {
            bucketList
                .ForEach(b => Console.WriteLine($"Bucket name: {b.BucketName},
 created on: {b.CreationDate}"));
        }

        public static async Task Main()
        {
            // The client uses the AWS Region of the default user.
            // If the Region where the buckets were created is different,
            // pass the Region to the client constructor. For example:
            // _s3Client = new AmazonS3Client(RegionEndpoint.USEast1);
            _s3Client = new AmazonS3Client();
            var response = await GetBuckets(_s3Client);
            DisplayBucketList(response.Buckets);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for .NET - Referenz für .

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::ListBuckets(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);

    auto outcome = client.ListBuckets();

    bool result = true;
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed with error: " << outcome.GetError() << std::endl;
        result = false;
    }
    else {
        std::cout << "Found " << outcome.GetResult().GetBuckets().size() << "
 buckets\n";
        for (auto &&b: outcome.GetResult().GetBuckets()) {
            std::cout << b.GetName() << std::endl;
        }
    }

    return result;
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl verwendet den `list-buckets` Befehl , um die Namen aller Ihrer Amazon S3-Buckets (alle Regionen) anzuzeigen:

```
aws s3api list-buckets --query "Buckets[].Name"
```

Die Abfrageoption filtert die Ausgabe von nur `list-buckets` auf die Bucket-Namen.

Weitere Informationen zu Buckets finden Sie unter Arbeiten mit Amazon S3-Buckets im Amazon S3-Entwicklerhandbuch.

- API-Details finden Sie unter ListBuckets in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// ListBuckets lists the buckets in the current account.
func (basics BucketBasics) ListBuckets() ([]types.Bucket, error) {
 result, err := basics.S3Client.ListBuckets(context.TODO(),
 &s3.ListBucketsInput{})
 var buckets []types.Bucket
 if err != nil {
  log.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
 } else {
  buckets = result.Buckets
 }
 return buckets, err
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for Go - Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Bucket;
import software.amazon.awssdk.services.s3.model.ListBucketsResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListBuckets {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        listAllBuckets(s3);

    }
    public static void listAllBuckets(S3Client s3) {
        ListBucketsResponse response = s3.listBuckets();
        List<Bucket> bucketList = response.buckets();
        for (Bucket bucket: bucketList) {
            System.out.println("Bucket name "+bucket.name());
        }
    }
```

```
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie die Buckets auf.

```
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new ListBucketsCommand({});

  try {
    const { Owner, Buckets } = await client.send(command);
    console.log(
      `${Owner.DisplayName} owns ${Buckets.length} bucket${
        Buckets.length === 1 ? "" : "s"
      }:`,
    );
    console.log(`${Buckets.map((b) => ` • ${b.Name}`).join("\n")}`);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for JavaScript - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    @staticmethod
    def list(s3_resource):
        """
        Get the buckets in all Regions for the current account.

        :param s3_resource: A Boto3 S3 resource. This is a high-level resource in
 Boto3
                            that contains collections and factory methods to
 create
                            other high-level S3 sub-resources.
        :return: The list of buckets.
        """
        try:
            buckets = list(s3_resource.buckets.all())
            logger.info("Got buckets: %s.", buckets)
```

```
        except ClientError:
            logger.exception("Couldn't get buckets.")
            raise
        else:
            return buckets
```

- Weitere API-Informationen finden Sie unter ListBuckets in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 resource actions.
class BucketListWrapper
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
  end

  # Lists buckets for the current account.
  #
  # @param count [Integer] The maximum number of buckets to list.
  def list_buckets(count)
    puts "Found these buckets:"
    @s3_resource.buckets.each do |bucket|
      puts "\t#{bucket.name}"
      count -= 1
      break if count.zero?
```

```
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't list buckets. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  wrapper = BucketListWrapper.new(Aws::S3::Resource.new)
  wrapper.list_buckets(25)
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter ListBuckets in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn show_buckets(strict: bool, client: &Client, region: &str) -> Result<(),
 Error> {
    let resp = client.list_buckets().send().await?;
    let buckets = resp.buckets();
    let num_buckets = buckets.len();

    let mut in_region = 0;

    for bucket in buckets {
        if strict {
            let r = client
```

```
                .get_bucket_location()
                .bucket(bucket.name().unwrap_or_default())
                .send()
                .await?;

            if r.location_constraint().unwrap().as_ref() == region {
                println!("{}", bucket.name().unwrap_or_default());
                in_region += 1;
            }
        } else {
            println!("{}", bucket.name().unwrap_or_default());
        }
    }

    println!();
    if strict {
        println!(
            "Found {} buckets in the {} region out of a total of {} buckets.",
            in_region, region, num_buckets
        );
    } else {
        println!("Found {} buckets in all regions.", num_buckets);
    }

    Ok(())
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der API-AWS Referenz zum -SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/// Return an array containing information about every available bucket.
///
/// - Returns: An array of ``S3ClientTypes.Bucket`` objects describing
///   each bucket.
public func getAllBuckets() async throws -> [S3ClientTypes.Bucket] {
    let output = try await client.listBuckets(input: ListBucketsInput())

    guard let buckets = output.buckets else {
        return []
    }
    return buckets
}
```

- Weitere API-Informationen finden Sie unter ListBuckets in der API-AWS Referenz zum -SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

In Bearbeitung befindliche mehrteilige Uploads in einen Amazon S3-Bucket mithilfe eines AWS -SDK auflisten

Die folgenden Codebeispiele zeigen, wie Sie in Bearbeitung befindliche mehrteilige Uploads in einen S3-Bucket auflisten.

CLI

AWS CLI

Der folgende Befehl listet alle aktiven mehrteiligen Uploads für einen Bucket mit dem Namen auf`my-bucket`:

```
aws s3api list-multipart-uploads --bucket my-bucket
```

Ausgabe:

```
{
    "Uploads": [
        {
            "Initiator": {
                "DisplayName": "username",
                "ID": "arn:aws:iam::0123456789012:user/username"
            },
            "Initiated": "2015-06-02T18:01:30.000Z",
            "UploadId":
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3
            "StorageClass": "STANDARD",
            "Key": "multipart/01",
            "Owner": {
                "DisplayName": "aws-account-name",
                "ID":
 "100719349fc3b6dcd7c820a124bf7aecd408092c3d7b51b38494939801fc248b"
            }
        }
    ],
    "CommonPrefixes": []
}
```

Bei laufenden mehrteiligen Uploads fallen Speicherkosten in Amazon S3 an. Schließen Sie einen aktiven mehrteiligen Upload ab oder brechen Sie ihn ab, um dessen Teile aus Ihrem Konto zu entfernen.

- API-Details finden Sie unter ListMultipartUploads in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsRequest;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsResponse;
import software.amazon.awssdk.services.s3.model.MultipartUpload;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class ListMultipartUploads {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The name of the Amazon S3 bucket where an in-
progress multipart upload is occurring.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();
        listUploads(s3, bucketName);
        s3.close();
    }
```

```
    public static void listUploads(S3Client s3, String bucketName) {
        try {
            ListMultipartUploadsRequest listMultipartUploadsRequest =
 ListMultipartUploadsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListMultipartUploadsResponse response =
 s3.listMultipartUploads(listMultipartUploadsRequest);
            List<MultipartUpload> uploads = response.uploads();
            for (MultipartUpload upload : uploads) {
                System.out.println("Upload in progress: Key = \"" + upload.key()
 + "\", id = " + upload.uploadId());
            }

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter ListMultipartUploads in der APIAWS SDK for Java 2.x -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Auflisten der Version von Objekten in einem Amazon S3-Bucket mithilfe eines AWS -SDK

Die folgenden Codebeispiele zeigen, wie Sie Objektversionen in einem S3-Bucket auflisten.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Mit versionierten Objekten arbeiten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example lists the versions of the objects in a version enabled
    /// Amazon Simple Storage Service (Amazon S3) bucket.
    /// </summary>
    public class ListObjectVersions
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";

            // If the AWS Region where your bucket is defined is different from
            // the AWS Region where the Amazon S3 bucket is defined, pass the
constant
            // for the AWS Region to the client constructor like this:
            //      var client = new AmazonS3Client(RegionEndpoint.USWest2);
            IAmazonS3 client = new AmazonS3Client();
            await GetObjectListWithAllVersionsAsync(client, bucketName);
        }

        /// <summary>
        /// This method lists all versions of the objects within an Amazon S3
        /// version enabled bucket.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// ListVersionsAsync.</param>
        /// <param name="bucketName">The name of the version enabled Amazon S3
bucket
```

```
        /// for which you want to list the versions of the contained objects.</
param>
        public static async Task GetObjectListWithAllVersionsAsync(IAmazonS3
 client, string bucketName)
        {
            try
            {
                // When you instantiate the ListVersionRequest, you can
                // optionally specify a key name prefix in the request
                // if you want a list of object versions of a specific object.

                // For this example we set a small limit in MaxKeys to return
                // a small list of versions.
                ListVersionsRequest request = new ListVersionsRequest()
                {
                    BucketName = bucketName,
                    MaxKeys = 2,
                };

                do
                {
                    ListVersionsResponse response = await
 client.ListVersionsAsync(request);

                    // Process response.
                    foreach (S3ObjectVersion entry in response.Versions)
                    {
                        Console.WriteLine($"key: {entry.Key} size:
 {entry.Size}");
                    }

                    // If response is truncated, set the marker to get the next
                    // set of keys.
                    if (response.IsTruncated)
                    {
                        request.KeyMarker = response.NextKeyMarker;
                        request.VersionIdMarker = response.NextVersionIdMarker;
                    }
                    else
                    {
                        request = null;
                    }
                }
                while (request != null);
```

```
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: '{ex.Message}'");
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter ListObjectVersions in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

Der folgende Befehl ruft Versionsinformationen für ein Objekt in einem Bucket mit dem Namen ab`my-bucket`:

```
aws s3api list-object-versions --bucket my-bucket --prefix index.html
```

Ausgabe:

```
{
    "DeleteMarkers": [
        {
            "Owner": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": true,
            "VersionId": "B2VsEK5saUNNHKcOAJj7hIE86RozToyq",
            "Key": "index.html",
            "LastModified": "2015-11-10T00:57:03.000Z"
        },
        {
            "Owner": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
```

```
            },
            "IsLatest": false,
            "VersionId": ".FLQEZscLIcfxSq.jsFJ.szUkmng2Yw6",
            "Key": "index.html",
            "LastModified": "2015-11-09T23:32:20.000Z"
        }
    ],
    "Versions": [
        {
            "LastModified": "2015-11-10T00:20:11.000Z",
            "VersionId": "Rb_l2T8UHDkFEwCgJjhlgPOZC0qJ.vpD",
            "ETag": "\"0622528de826c0df5db1258a23b80be5\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 38
        },
        {
            "LastModified": "2015-11-09T23:26:41.000Z",
            "VersionId": "rasWWGpgk9E4s0LyTJgusGeRQKLVIAFf",
            "ETag": "\"06225825b8028de826c0df5db1a23be5\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 38
        },
        {
            "LastModified": "2015-11-09T22:50:50.000Z",
            "VersionId": "null",
            "ETag": "\"d1f45267a863c8392e07d24dd592f1b9\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
```

```
                "ID":
 "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 533823
        }
    ]
}
```

- API-Details finden Sie unter ListObjectVersions in der AWS CLI -Befehlsreferenz.

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn show_versions(client: &Client, bucket: &str) -> Result<(), Error> {
    let resp = client.list_object_versions().bucket(bucket).send().await?;

    for version in resp.versions() {
        println!("{}", version.key().unwrap_or_default());
        println!("  version ID: {}", version.version_id().unwrap_or_default());
        println!();
    }

    Ok(())
}
```

- Weitere API-Informationen finden Sie unter ListObjectVersions in der API-AWS Referenz zum -SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Auflisten von Objekten in einem Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird veranschaulicht, wie Sie Objekte in einem S3 Bucket auflisten.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Buckets und Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Shows how to list the objects in an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket for which to list
        /// the contents.</param>
        /// <returns>A boolean value indicating the success or failure of the
        /// copy operation.</returns>
        public static async Task<bool> ListBucketContentsAsync(IAmazonS3 client,
string bucketName)
        {
            try
            {
                var request = new ListObjectsV2Request
                {
                    BucketName = bucketName,
                    MaxKeys = 5,
                };

                Console.WriteLine("--------------------------------------");
```

```
                Console.WriteLine($"Listing the contents of {bucketName}:");
                Console.WriteLine("--------------------------------------");

                ListObjectsV2Response response;

                do
                {
                    response = await client.ListObjectsV2Async(request);

                    response.S3Objects
                        .ForEach(obj => Console.WriteLine($"{obj.Key,-35}
{obj.LastModified.ToShortDateString(),10}{obj.Size,10}"));

                    // If the response is truncated, set the request
 ContinuationToken
                    // from the NextContinuationToken property of the response.
                    request.ContinuationToken = response.NextContinuationToken;
                }
                while (response.IsTruncated);

                return true;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error encountered on server.
 Message:'{ex.Message}' getting list of objects.");
                return false;
            }
        }
```

Listen Sie Objekte mit einem Paginator auf.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// The following example lists objects in an Amazon Simple Storage
    /// Service (Amazon S3) bucket.
    /// </summary>
```

```
    public class ListObjectsPaginator
    {
        private const string BucketName = "doc-example-bucket";

        public static async Task Main()
        {
            IAmazonS3 s3Client = new AmazonS3Client();

            Console.WriteLine($"Listing the objects contained in {BucketName}:
\n");
            await ListingObjectsAsync(s3Client, BucketName);
        }

        /// <summary>
        /// This method uses a paginator to retrieve the list of objects in an
        /// an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the S3 bucket whose objects
        /// you want to list.</param>
        public static async Task ListingObjectsAsync(IAmazonS3 client, string
 bucketName)
        {
            var listObjectsV2Paginator = client.Paginators.ListObjectsV2(new
 ListObjectsV2Request
            {
                BucketName = bucketName,
            });

            await foreach (var response in listObjectsV2Paginator.Responses)
            {
                Console.WriteLine($"HttpStatusCode: {response.HttpStatusCode}");
                Console.WriteLine($"Number of Keys: {response.KeyCount}");
                foreach (var entry in response.S3Objects)
                {
                    Console.WriteLine($"Key = {entry.Key} Size = {entry.Size}");
                }
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#       $1 - The name of the bucket.
#
# Returns:
#       The list of files in text format.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function list_items_in_bucket() {
  local bucket_name=$1
  local response
```

```
  response=$(aws s3api list-objects \
    --bucket "$bucket_name" \
    --output text \
    --query 'Contents[].{Key: Key, Size: Size}')

  # shellcheck disable=SC2181
  if [[ ${?} -eq 0 ]]; then
    echo "$response"
  else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
  fi
}
```

- API-Details finden Sie unter ListObjectsV2 in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::ListObjects(const Aws::String &bucketName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::ListObjectsRequest request;
    request.WithBucket(bucketName);

    auto outcome = s3_client.ListObjects(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: ListObjects: " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        Aws::Vector<Aws::S3::Model::Object> objects =
                outcome.GetResult().GetContents();

        for (Aws::S3::Model::Object &object: objects) {
            std::cout << object.GetKey() << std::endl;
        }
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for C++ -Referenz für .

CLI

AWS CLI

Im folgenden Beispiel wird der `list-objects` Befehl verwendet, um die Namen aller Objekte im angegebenen Bucket anzuzeigen:

```
aws s3api list-objects --bucket text-content --query 'Contents[].{Key: Key, Size:
 Size}'
```

Das Beispiel verwendet das Argument `--query` , um die Ausgabe von auf den Schlüsselwert und die Größe für jedes Objekt `list-objects` zu filtern

Weitere Informationen zu Objekten finden Sie unter Arbeiten mit Amazon S3-Objekten im Amazon S3-Entwicklerhandbuch.

- API-Details finden Sie unter ListObjectsV2 in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// ListObjects lists the objects in a bucket.
func (basics BucketBasics) ListObjects(bucketName string) ([]types.Object, error)
 {
 result, err := basics.S3Client.ListObjectsV2(context.TODO(),
 &s3.ListObjectsV2Input{
  Bucket: aws.String(bucketName),
 })
 var contents []types.Object
 if err != nil {
  log.Printf("Couldn't list objects in bucket %v. Here's why: %v\n", bucketName,
 err)
 } else {
  contents = result.Contents
 }
 return contents, err
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for Go - Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.S3Object;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class ListObjects {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket from which objects are
 read.\s
```

```
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBucketObjects(s3, bucketName);
        s3.close();
    }

    public static void listBucketObjects(S3Client s3, String bucketName) {
        try {
            ListObjectsRequest listObjects = ListObjectsRequest
                    .builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.print("\n The name of the key is " + myValue.key());
                System.out.print("\n The object is " + calKb(myValue.size()) + "
 KBs");
                System.out.print("\n The owner is " + myValue.owner());
            }

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // convert bytes to kbs.
    private static long calKb(Long val) {
        return val / 1024;
    }
}
```

Listet Objekte mithilfe der Paginierung auf.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.paginators.ListObjectsV2Iterable;

public class ListObjectsPaginated {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket from which objects are
 read.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBucketObjects(s3, bucketName);
        s3.close();
    }

    public static void listBucketObjects(S3Client s3, String bucketName) {
        try {
            ListObjectsV2Request listReq = ListObjectsV2Request.builder()
                    .bucket(bucketName)
                    .maxKeys(1)
                    .build();
```

```
            ListObjectsV2Iterable listRes = s3.listObjectsV2Paginator(listReq);
            listRes.stream()
                    .flatMap(r -> r.contents().stream())
                    .forEach(content -> System.out.println(" Key: " +
 content.key() + " size = " + content.size()));

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie alle Objekte in Ihrem Bucket auf. Wenn es mehr als ein Objekt gibt, IsTruncated NextContinuationToken wird verwendet, um über die vollständige Liste zu iterieren.

```
import {
  S3Client,
  // This command supersedes the ListObjectsCommand and is the recommended way to
 list objects.
  ListObjectsV2Command,
} from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
```

```
  const command = new ListObjectsV2Command({
    Bucket: "my-bucket",
    // The default and maximum number of keys returned is 1000. This limits it to
    // one for demonstration purposes.
    MaxKeys: 1,
  });

  try {
    let isTruncated = true;

    console.log("Your bucket contains the following objects:\n");
    let contents = "";

    while (isTruncated) {
      const { Contents, IsTruncated, NextContinuationToken } =
        await client.send(command);
      const contentsList = Contents.map((c) => ` • ${c.Key}`).join("\n");
      contents += contentsList + "\n";
      isTruncated = IsTruncated;
      command.input.ContinuationToken = NextContinuationToken;
    }
    console.log(contents);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for JavaScript -Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listBucketObjects(bucketName: String) {
    val request = ListObjectsRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->

        val response = s3.listObjects(request)
        response.contents?.forEach { myObject ->
            println("The name of the key is ${myObject.key}")
            println("The object is ${myObject.size?.let { calKb(it) }} KBs")
            println("The owner is ${myObject.owner}")
        }
    }
}

private fun calKb(intValue: Long): Long {
    return intValue / 1024
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum -SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie Objekte in einem Bucket auf.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $contents = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
```

```
            ]);
            echo "The contents of your bucket are: \n";
            foreach ($contents['Contents'] as $content) {
                echo $content['Key'] . "\n";
            }
        } catch (Exception $exception) {
            echo "Failed to list objects in $this->bucketName with error: " .
$exception->getMessage();
            exit("Please fix error with listing objects before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    @staticmethod
    def list(bucket, prefix=None):
        """
```

```
        Lists the objects in a bucket, optionally filtered by a prefix.

        :param bucket: The bucket to query. This is a Boto3 Bucket resource.
        :param prefix: When specified, only objects that start with this prefix
are listed.
        :return: The list of objects.
        """
        try:
            if not prefix:
                objects = list(bucket.objects.all())
            else:
                objects = list(bucket.objects.filter(Prefix=prefix))
            logger.info(
                "Got objects %s from bucket '%s'", [o.key for o in objects],
bucket.name
            )
        except ClientError:
            logger.exception("Couldn't get objects for bucket '%s'.",
bucket.name)
            raise
        else:
            return objects
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
```

```
class BucketListObjectsWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
  def initialize(bucket)
    @bucket = bucket
  end

  # Lists object in a bucket.
  #
  # @param max_objects [Integer] The maximum number of objects to list.
  # @return [Integer] The number of objects listed.
  def list_objects(max_objects)
    count = 0
    puts "The objects in #{@bucket.name} are:"
    @bucket.objects.each do |obj|
      puts "\t#{obj.key}"
      count += 1
      break if count == max_objects
    end
    count
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't list objects in bucket #{bucket.name}. Here's why:
 #{e.message}"
    0
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"

  wrapper = BucketListObjectsWrapper.new(Aws::S3::Bucket.new(bucket_name))
  count = wrapper.list_objects(25)
  puts "Listed #{count} objects."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn list_objects(client: &Client, bucket: &str) -> Result<(), Error> {
    let mut response = client
        .list_objects_v2()
        .bucket(bucket.to_owned())
        .max_keys(10) // In this example, go 10 at a time.
        .into_paginator()
        .send();

    while let Some(result) = response.next().await {
        match result {
            Ok(output) => {
                for object in output.contents() {
                    println!(" - {}", object.key().unwrap_or("Unknown"));
                }
            }
            Err(err) => {
                eprintln!("{err:?}")
            }
        }
    }

    Ok(())
}
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum - SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    TRY.
        oo_result = lo_s3->listobjectsv2(         " oo_result is returned for
testing purposes. "
          iv_bucket = iv_bucket_name
        ).
        MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum -SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    public func listBucketFiles(bucket: String) async throws -> [String] {
        let input = ListObjectsV2Input(
            bucket: bucket
        )
        let output = try await client.listObjectsV2(input: input)
        var names: [String] = []

        guard let objList = output.contents else {
            return []
        }

        for obj in objList {
            if let objName = obj.key {
                names.append(objName)
            }
        }

        return names
    }
```

- Weitere API-Informationen finden Sie unter ListObjectsV2 in der AWS API-Referenz zum - SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Wiederherstellen einer archivierten Kopie eines Objekts in einem Amazon S3-Bucket mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie eine archivierte Kopie eines Objekts in einem S3-Bucket wiederherstellen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    using System;
    using System.Threading.Tasks;
    using Amazon;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to restore an archived object in an Amazon
    /// Simple Storage Service (Amazon S3) bucket.
    /// </summary>
    public class RestoreArchivedObject
    {
        public static void Main()
        {
            string bucketName = "doc-example-bucket";
            string objectKey = "archived-object.txt";

            // Specify your bucket region (an example region is shown).
            RegionEndpoint bucketRegion = RegionEndpoint.USWest2;

            IAmazonS3 client = new AmazonS3Client(bucketRegion);
            RestoreObjectAsync(client, bucketName, objectKey).Wait();
        }

        /// <summary>
        /// This method restores an archived object from an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// RestoreObjectAsync.</param>
        /// <param name="bucketName">A string representing the name of the
        /// bucket where the object was located before it was archived.</param>
```

```
        /// <param name="objectKey">A string representing the name of the
        /// archived object to restore.</param>
        public static async Task RestoreObjectAsync(IAmazonS3 client, string
bucketName, string objectKey)
        {
            try
            {
                var restoreRequest = new RestoreObjectRequest
                {
                    BucketName = bucketName,
                    Key = objectKey,
                    Days = 2,
                };
                RestoreObjectResponse response = await
client.RestoreObjectAsync(restoreRequest);

                // Check the status of the restoration.
                await CheckRestorationStatusAsync(client, bucketName, objectKey);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine($"Error: {amazonS3Exception.Message}");
            }
        }

        /// <summary>
        /// This method retrieves the status of the object's restoration.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// GetObjectMetadataAsync.</param>
        /// <param name="bucketName">A string representing the name of the Amazon
        /// S3 bucket which contains the archived object.</param>
        /// <param name="objectKey">A string representing the name of the
        /// archived object you want to restore.</param>
        public static async Task CheckRestorationStatusAsync(IAmazonS3 client,
string bucketName, string objectKey)
        {
            GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
            };
```

```
            GetObjectMetadataResponse response = await
client.GetObjectMetadataAsync(metadataRequest);

            var restStatus = response.RestoreInProgress ? "in-progress" :
"finished or failed";
            Console.WriteLine($"Restoration status: {restStatus}");
        }
    }
```

- Weitere API-Informationen finden Sie unter RestoreObject in der APIAWS SDK for .NET - Referenz für .

CLI

AWS CLI

So erstellen Sie eine Wiederherstellungsanforderung für ein Objekt

Im folgenden `restore-object` Beispiel wird das angegebene Amazon S3-Glacier-Objekt für den Bucket 10 Tage `my-glacier-bucket` lang wiederhergestellt.

```
aws s3api restore-object \
    --bucket my-glacier-bucket \
    --key doc1.rtf \
    --restore-request Days=10
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- API-Details finden Sie unter RestoreObject in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.RestoreRequest;
import software.amazon.awssdk.services.s3.model.GlacierJobParameters;
import software.amazon.awssdk.services.s3.model.RestoreObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Tier;

/*
 *  For more information about restoring an object, see "Restoring an archived
 object" at
 *  https://docs.aws.amazon.com/AmazonS3/latest/userguide/restoring-objects.html
 *
 *  Before running this Java V2 code example, set up your development
 environment, including your credentials.
 *
 *  For more information, see the following documentation topic:
 *
 *  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class RestoreObject {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName> <expectedBucketOwner>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name of an object with a Storage class
 value of Glacier.\s
                    expectedBucketOwner - The account that owns the bucket (you
 can obtain this value from the AWS Management Console).\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
```

```
        String expectedBucketOwner = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        restoreS3Object(s3, bucketName, keyName, expectedBucketOwner);
        s3.close();
    }

    public static void restoreS3Object(S3Client s3, String bucketName, String
 keyName, String expectedBucketOwner) {
        try {
            RestoreRequest restoreRequest = RestoreRequest.builder()
                    .days(10)

 .glacierJobParameters(GlacierJobParameters.builder().tier(Tier.STANDARD).build())
                    .build();

            RestoreObjectRequest objectRequest = RestoreObjectRequest.builder()
                    .expectedBucketOwner(expectedBucketOwner)
                    .bucket(bucketName)
                    .key(keyName)
                    .restoreRequest(restoreRequest)
                    .build();

            s3.restoreObject(objectRequest);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter RestoreObject in der APIAWS SDK for Java 2.x -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Festlegen einer neuen ACL für einen Amazon S3-Bucket mithilfe eines - AWS SDK

Die folgenden Codebeispiele veranschaulichen, wie Sie eine neue Zugriffssteuerungsliste (Access Control List, ACL) für einen S3 Bucket festlegen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Zugriffssteuerungslisten (ACL)

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Creates an Amazon S3 bucket with an ACL to control access to the
        /// bucket and the objects stored in it.
        /// </summary>
        /// <param name="client">The initialized client object used to create
        /// an Amazon S3 bucket, with an ACL applied to the bucket.
        /// </param>
        /// <param name="region">The AWS Region where the bucket will be
created.</param>
        /// <param name="newBucketName">The name of the bucket to create.</param>
        /// <returns>A boolean value indicating success or failure.</returns>
        public static async Task<bool> CreateBucketUseCannedACLAsync(IAmazonS3
client, S3Region region, string newBucketName)
        {
            try
            {
                // Create a new Amazon S3 bucket with Canned ACL.
                var putBucketRequest = new PutBucketRequest()
                {
```

```
                    BucketName = newBucketName,
                    BucketRegion = region,
                    CannedACL = S3CannedACL.LogDeliveryWrite,
                };

                PutBucketResponse putBucketResponse = await
client.PutBucketAsync(putBucketRequest);

                return putBucketResponse.HttpStatusCode ==
System.Net.HttpStatusCode.OK;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Amazon S3 error: {ex.Message}");
            }

            return false;
        }
```

- Weitere API-Informationen finden Sie unter PutBucketAcl in der APIAWS SDK for .NET - Referenz für .

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::PutBucketAcl(const Aws::String &bucketName,
                              const Aws::String &ownerID,
                              const Aws::String &granteePermission,
                              const Aws::String &granteeType,
                              const Aws::String &granteeID,
                              const Aws::Client::ClientConfiguration
 &clientConfig,
```

```
                            const Aws::String &granteeDisplayName,
                            const Aws::String &granteeEmailAddress,
                            const Aws::String &granteeURI) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::Owner owner;
    owner.SetID(ownerID);

    Aws::S3::Model::Grantee grantee;
    grantee.SetType(SetGranteeType(granteeType));

    if (!granteeEmailAddress.empty()) {
        grantee.SetEmailAddress(granteeEmailAddress);
    }

    if (!granteeID.empty()) {
        grantee.SetID(granteeID);
    }

    if (!granteeDisplayName.empty()) {
        grantee.SetDisplayName(granteeDisplayName);
    }

    if (!granteeURI.empty()) {
        grantee.SetURI(granteeURI);
    }

    Aws::S3::Model::Grant grant;
    grant.SetGrantee(grantee);
    grant.SetPermission(SetGranteePermission(granteePermission));

    Aws::Vector<Aws::S3::Model::Grant> grants;
    grants.push_back(grant);

    Aws::S3::Model::AccessControlPolicy acp;
    acp.SetOwner(owner);
    acp.SetGrants(grants);

    Aws::S3::Model::PutBucketAclRequest request;
    request.SetAccessControlPolicy(acp);
    request.SetBucket(bucketName);

    Aws::S3::Model::PutBucketAclOutcome outcome =
            s3_client.PutBucketAcl(request);
```

```
    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &error = outcome.GetError();

        std::cerr << "Error: PutBucketAcl: " << error.GetExceptionName()
                  << " - " << error.GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully added an ACL to the bucket '" << bucketName
                  << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \sa SetGranteePermission()
 \param access Human readable string.
*/

Aws::S3::Model::Permission SetGranteePermission(const Aws::String &access) {
    if (access == "FULL_CONTROL")
        return Aws::S3::Model::Permission::FULL_CONTROL;
    if (access == "WRITE")
        return Aws::S3::Model::Permission::WRITE;
    if (access == "READ")
        return Aws::S3::Model::Permission::READ;
    if (access == "WRITE_ACP")
        return Aws::S3::Model::Permission::WRITE_ACP;
    if (access == "READ_ACP")
        return Aws::S3::Model::Permission::READ_ACP;
    return Aws::S3::Model::Permission::NOT_SET;
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \sa SetGranteeType()
 \param type Human readable string.
*/

Aws::S3::Model::Type SetGranteeType(const Aws::String &type) {
```

```
    if (type == "Amazon customer by email")
        return Aws::S3::Model::Type::AmazonCustomerByEmail;
    if (type == "Canonical user")
        return Aws::S3::Model::Type::CanonicalUser;
    if (type == "Group")
        return Aws::S3::Model::Type::Group;
    return Aws::S3::Model::Type::NOT_SET;
}
```

- Weitere API-Informationen finden Sie unter PutBucketAcl in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

In diesem Beispiel `full control` werden zwei AWS Benutzern (user1@example.com und user2@example.com) und allen Benutzern die `read` Berechtigung erteilt:

```
aws s3api put-bucket-acl --bucket MyBucket --grant-full-control
 emailaddress=user1@example.com,emailaddress=user2@example.com --grant-read
 uri=http://acs.amazonaws.com/groups/global/AllUsers
```

Weitere Informationen zu benutzerdefinierten ACLs finden Sie unter http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html (die S3api-ACL-Befehle, wie z. B. `put-bucket-acl`, verwenden dieselbe Kurznotation für Argumente).

- API-Details finden Sie unter PutBucketAcl in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import java.util.ArrayList;
import java.util.List;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.services.s3.model.Permission;
import software.amazon.awssdk.services.s3.model.Grant;
import software.amazon.awssdk.services.s3.model.AccessControlPolicy;
import software.amazon.awssdk.services.s3.model.Type;
import software.amazon.awssdk.services.s3.model.PutBucketAclRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SetAcl {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <bucketName> <id>\s

                Where:
                  bucketName - The Amazon S3 bucket to grant permissions on.\s
                  id - The ID of the owner of this bucket (you can get this value
 from the AWS Management Console).
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String id = args[1];
        System.out.format("Setting access \n");
        System.out.println(" in bucket: " + bucketName);
```

```
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setBucketAcl(s3, bucketName, id);
        System.out.println("Done!");
        s3.close();
    }

    public static void setBucketAcl(S3Client s3, String bucketName, String id) {
        try {
            Grant ownerGrant = Grant.builder()
                    .grantee(builder -> builder.id(id)
                            .type(Type.CANONICAL_USER))
                    .permission(Permission.FULL_CONTROL)
                    .build();

            List<Grant> grantList2 = new ArrayList<>();
            grantList2.add(ownerGrant);

            AccessControlPolicy acl = AccessControlPolicy.builder()
                    .owner(builder -> builder.id(id))
                    .grants(grantList2)
                    .build();

            PutBucketAclRequest putAclReq = PutBucketAclRequest.builder()
                    .bucket(bucketName)
                    .accessControlPolicy(acl)
                    .build();

            s3.putBucketAcl(putAclReq);

        } catch (S3Exception e) {
            e.printStackTrace();
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter PutBucketAcl in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Legen Sie die Bucket-ACL fest.

```
import {
  PutBucketAclCommand,
  GetBucketAclCommand,
  S3Client,
} from "@aws-sdk/client-s3";

const client = new S3Client({});

// Most Amazon S3 use cases don't require the use of access control lists (ACLs).
// We recommend that you disable ACLs, except in unusual circumstances where
// you need to control access for each object individually.
// Consider a policy instead. For more information see https://
docs.aws.amazon.com/AmazonS3/latest/userguide/bucket-policies.html.
export const main = async () => {
  // Grant a user READ access to a bucket.
  const command = new PutBucketAclCommand({
    Bucket: "test-bucket",
    AccessControlPolicy: {
      Grants: [
        {
          Grantee: {
            // The canonical ID of the user. This ID is an obfuscated form of
 your AWS account number.
            // It's unique to Amazon S3 and can't be found elsewhere.
            // For more information, see https://docs.aws.amazon.com/AmazonS3/
latest/userguide/finding-canonical-user-id.html.
            ID: "canonical-id-1",
            Type: "CanonicalUser",
          },
          // One of FULL_CONTROL | READ | WRITE | READ_ACP | WRITE_ACP
```

```
          // https://docs.aws.amazon.com/AmazonS3/latest/API/
API_Grant.html#AmazonS3-Type-Grant-Permission
          Permission: "FULL_CONTROL",
        },
      ],
      Owner: {
        ID: "canonical-id-2",
      },
    },
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter PutBucketAcl in der APIAWS SDK for JavaScript -Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun setBucketAcl(bucketName: String, idVal: String) {
    val myGrant = Grantee {
        id = idVal
        type = Type.CanonicalUser
    }

    val ownerGrant = Grant {
```

```
        grantee = myGrant
        permission = Permission.FullControl
    }

    val grantList = mutableListOf<Grant>()
    grantList.add(ownerGrant)

    val ownerOb = Owner {
        id = idVal
    }

    val acl = AccessControlPolicy {
        owner = ownerOb
        grants = grantList
    }

    val request = PutBucketAclRequest {
        bucket = bucketName
        accessControlPolicy = acl
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.putBucketAcl(request)
        println("An ACL was successfully set on $bucketName")
    }
}
```

- Weitere API-Informationen finden Sie unter PutBucketAcl in der API-AWS Referenz zum - SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def grant_log_delivery_access(self):
        """
        Grant the AWS Log Delivery group write access to the bucket so that
        Amazon S3 can deliver access logs to the bucket. This is the only
 recommended
        use of an S3 bucket ACL.
        """
        try:
            acl = self.bucket.Acl()
            # Putting an ACL overwrites the existing ACL. If you want to preserve
            # existing grants, append new grants to the list of existing grants.
            grants = acl.grants if acl.grants else []
            grants.append(
                {
                    "Grantee": {
                        "Type": "Group",
                        "URI": "http://acs.amazonaws.com/groups/s3/LogDelivery",
                    },
                    "Permission": "WRITE",
                }
            )
            acl.put(AccessControlPolicy={"Grants": grants, "Owner": acl.owner})
            logger.info("Granted log delivery access to bucket '%s'",
 self.bucket.name)
        except ClientError:
            logger.exception("Couldn't add ACL to bucket '%s'.",
 self.bucket.name)
            raise
```

- Weitere API-Informationen finden Sie unter PutBucketAcl in der AWS API-Referenz zum - SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Festlegen der ACL eines Amazon S3-Objekts mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Zugriffssteuerungsliste (ACL) eines S3-Objekts festlegen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Verwalten von Zugriffssteuerungslisten (ACL)

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::PutObjectAcl(const Aws::String &bucketName,
                              const Aws::String &objectKey,
                              const Aws::String &ownerID,
                              const Aws::String &granteePermission,
                              const Aws::String &granteeType,
                              const Aws::String &granteeID,
                              const Aws::Client::ClientConfiguration
 &clientConfig,
                              const Aws::String &granteeDisplayName,
                              const Aws::String &granteeEmailAddress,
                              const Aws::String &granteeURI) {
    Aws::S3::S3Client s3_client(clientConfig);
```

```
    Aws::S3::Model::Owner owner;
    owner.SetID(ownerID);

    Aws::S3::Model::Grantee grantee;
    grantee.SetType(SetGranteeType(granteeType));

    if (!granteeEmailAddress.empty()) {
        grantee.SetEmailAddress(granteeEmailAddress);
    }

    if (!granteeID.empty()) {
        grantee.SetID(granteeID);
    }

    if (!granteeDisplayName.empty()) {
        grantee.SetDisplayName(granteeDisplayName);
    }

    if (!granteeURI.empty()) {
        grantee.SetURI(granteeURI);
    }

    Aws::S3::Model::Grant grant;
    grant.SetGrantee(grantee);
    grant.SetPermission(SetGranteePermission(granteePermission));

    Aws::Vector<Aws::S3::Model::Grant> grants;
    grants.push_back(grant);

    Aws::S3::Model::AccessControlPolicy acp;
    acp.SetOwner(owner);
    acp.SetGrants(grants);

    Aws::S3::Model::PutObjectAclRequest request;
    request.SetAccessControlPolicy(acp);
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::PutObjectAclOutcome outcome =
            s3_client.PutObjectAcl(request);

    if (!outcome.IsSuccess()) {
        auto error = outcome.GetError();
        std::cerr << "Error: PutObjectAcl: " << error.GetExceptionName()
```

```
                  << " - " << error.GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully added an ACL to the object '" << objectKey
                  << "' in the bucket '" << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \sa SetGranteePermission()
 \param access Human readable string.
*/

Aws::S3::Model::Permission SetGranteePermission(const Aws::String &access) {
    if (access == "FULL_CONTROL")
        return Aws::S3::Model::Permission::FULL_CONTROL;
    if (access == "WRITE")
        return Aws::S3::Model::Permission::WRITE;
    if (access == "READ")
        return Aws::S3::Model::Permission::READ;
    if (access == "WRITE_ACP")
        return Aws::S3::Model::Permission::WRITE_ACP;
    if (access == "READ_ACP")
        return Aws::S3::Model::Permission::READ_ACP;
    return Aws::S3::Model::Permission::NOT_SET;
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \sa SetGranteeType()
 \param type Human readable string.
*/

Aws::S3::Model::Type SetGranteeType(const Aws::String &type) {
    if (type == "Amazon customer by email")
        return Aws::S3::Model::Type::AmazonCustomerByEmail;
    if (type == "Canonical user")
        return Aws::S3::Model::Type::CanonicalUser;
    if (type == "Group")
```

```
        return Aws::S3::Model::Type::Group;
    return Aws::S3::Model::Type::NOT_SET;
}
```

- Weitere API-Informationen finden Sie unter PutObjectAcl in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der folgende Befehl erteilt `full control` zwei AWS Benutzern (user1@example.com und user2@example.com) und allen Benutzern die `read` Berechtigung:

```
aws s3api put-object-acl --bucket MyBucket --key file.txt --grant-full-control
 emailaddress=user1@example.com,emailaddress=user2@example.com --grant-read
 uri=http://acs.amazonaws.com/groups/global/AllUsers
```

Weitere Informationen zu benutzerdefinierten ACLs finden Sie unter http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html (die s3api-ACL-Befehle, z. B. `put-object-acl`, verwenden dieselbe Kurznotation für Argumente).

- API-Details finden Sie unter PutObjectAcl in der AWS CLI -Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""
```

```
    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def put_acl(self, email):
        """
        Applies an ACL to the object that grants read access to an AWS user
identified
        by email address.

        :param email: The email address of the user to grant access.
        """
        try:
            acl = self.object.Acl()
            # Putting an ACL overwrites the existing ACL, so append new grants
            # if you want to preserve existing grants.
            grants = acl.grants if acl.grants else []
            grants.append(
                {
                    "Grantee": {"Type": "AmazonCustomerByEmail", "EmailAddress":
email},
                    "Permission": "READ",
                }
            )
            acl.put(AccessControlPolicy={"Grants": grants, "Owner": acl.owner})
            logger.info("Granted read access to %s.", email)
        except ClientError:
            logger.exception("Couldn't add ACL to object '%s'.", self.object.key)
            raise
```

- Weitere API-Informationen finden Sie unter PutObjectAcl in der AWS API-Referenz zum - SDK für Python (Boto3).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Festlegen des Standardaufbewahrungszeitraums eines Amazon S3-Buckets mithilfe eines - AWS SDK

Das folgende Codebeispiel zeigt, wie der Standardaufbewahrungszeitraum eines S3-Buckets festgelegt wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Sperren von Amazon S3-Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Set or modify a retention period on an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket to modify.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date for retention until.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyBucketDefaultRetention(string bucketName, bool
enableObjectLock, ObjectLockRetentionMode retention, DateTime retainUntilDate)
    {
        var enabledString = enableObjectLock ? "Enabled" : "Disabled";
        var timeDifference = retainUntilDate.Subtract(DateTime.Now);
        try
        {
            // First, enable Versioning on the bucket.
```

```
            await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
            {
                BucketName = bucketName,
                VersioningConfig = new S3BucketVersioningConfig()
                {
                    EnableMfaDelete = false,
                    Status = VersionStatus.Enabled
                }
            });

            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled(enabledString),
                    Rule = new ObjectLockRule()
                    {
                        DefaultRetention = new DefaultRetention()
                        {
                            Mode = retention,
                            Days = timeDifference.Days // Can be specified in
days or years but not both.
                        }
                    }
                }
            };

            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded a default retention to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying object lock: '{ex.Message}'");
            return false;
        }
    }
```

- Weitere API-Informationen finden Sie unter PutObjectLockConfiguration in der APIAWS SDK for .NET -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Festlegen der Konfiguration für die rechtliche Aufbewahrungsfrist eines Amazon S3-Objekts mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Konfiguration für die rechtliche Aufbewahrungsfrist eines S3-Objekts festlegen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Sperren von Amazon S3-Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/// <summary>
/// Set or modify a legal hold on an object in an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The key of the object.</param>
/// <param name="holdStatus">The On or Off status for the legal hold.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ModifyObjectLegalHold(string bucketName,
    string objectKey, ObjectLockLegalHoldStatus holdStatus)
{
    try
```

```
        {
            var request = new PutObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                LegalHold = new ObjectLockLegalHold()
                {
                    Status = holdStatus
                }
            };

            var response = await _amazonS3.PutObjectLegalHoldAsync(request);
            Console.WriteLine($"\tModified legal hold for {objectKey} in
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying legal hold: '{ex.Message}'");
            return false;
        }
    }
```

- Weitere API-Informationen finden Sie unter PutObjectLegalHold in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

So wenden Sie eine rechtliche Aufbewahrungsfrist auf ein Objekt an

Im folgenden `put-object-legal-hold` Beispiel wird eine rechtliche Aufbewahrungsfrist für das Objekt festgelegt`doc1.rtf`.

```
aws s3api put-object-legal-hold \
    --bucket my-bucket-with-object-lock \
    --key doc1.rtf \
    --legal-hold Status=ON
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- API-Details finden Sie unter PutObjectLegalHold in der AWS CLI -Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Festlegen der Objektsperrkonfiguration eines Amazon S3-Buckets mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie die Objektsperrkonfiguration eines vorhandenen S3-Buckets festlegen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Sperren von Amazon S3-Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Enable object lock on an existing bucket.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to modify.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> EnableObjectLockOnBucket(string bucketName)
    {
        try
        {
            // First, enable Versioning on the bucket.
            await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
```

```
            {
                BucketName = bucketName,
                VersioningConfig = new S3BucketVersioningConfig()
                {
                    EnableMfaDelete = false,
                    Status = VersionStatus.Enabled
                }
            });

            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled("Enabled"),
                },
            };

            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded an object lock policy to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error modifying object lock: '{ex.Message}'");
            return false;
        }
    }
```

- Weitere API-Informationen finden Sie unter PutObjectLockConfiguration in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

So legen Sie eine Objektsperrenkonfiguration für einen Bucket fest

Im folgenden `put-object-lock-configuration` Beispiel wird eine 50-tägige Objektsperre für den angegebenen Bucket festgelegt.

```
aws s3api put-object-lock-configuration \
    --bucket my-bucket-with-object-lock \
    --object-lock-configuration '{ "ObjectLockEnabled": "Enabled", "Rule":
 { "DefaultRetention": { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- API-Details finden Sie unter PutObjectLockConfiguration in der AWS CLI -Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Festlegen des Aufbewahrungszeitraums eines Amazon S3-Objekts mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie den Aufbewahrungszeitraum eines S3-Objekts festlegen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Sperren von Amazon S3-Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Set or modify a retention period on an object in an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
```

```
    /// <param name="objectKey">The key of the object.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date retention expires.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyObjectRetentionPeriod(string bucketName,
        string objectKey, ObjectLockRetentionMode retention, DateTime
retainUntilDate)
    {
        try
        {
            var request = new PutObjectRetentionRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                Retention = new ObjectLockRetention()
                {
                    Mode = retention,
                    RetainUntilDate = retainUntilDate
                }
            };

            var response = await _amazonS3.PutObjectRetentionAsync(request);
            Console.WriteLine($"\tSet retention for {objectKey} in {bucketName}
until {retainUntilDate:d}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying retention period:
'{ex.Message}'");
            return false;
        }
    }
```

- Weitere API-Informationen finden Sie unter PutObjectRetention in der APIAWS SDK for .NET -Referenz für .

CLI

AWS CLI

So legen Sie eine Objektaufbewahrungskonfiguration für ein Objekt fest

Im folgenden `put-object-retention` Beispiel wird eine Objektaufbewahrungskonfiguration für das angegebene Objekt bis zum 2025-01-01 festgelegt.

```
aws s3api put-object-retention \
    --bucket my-bucket-with-object-lock \
    --key doc1.rtf \
    --retention '{ "Mode": "GOVERNANCE", "RetainUntilDate":
 "2025-01-01T00:00:00" }'
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- API-Details finden Sie unter PutObjectRetention in der AWS CLI -Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Festlegen der Website-Konfiguration für einen Amazon S3-Bucket mithilfe eines - AWS SDK

Das folgende Beispiel veranschaulicht, wie Sie die Website-Konfiguration für einen S3 Bucket festlegen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
                // Put the website configuration.
                PutBucketWebsiteRequest putRequest = new
PutBucketWebsiteRequest()
                {
                    BucketName = bucketName,
                    WebsiteConfiguration = new WebsiteConfiguration()
                    {
```

```
                        IndexDocumentSuffix = indexDocumentSuffix,
                        ErrorDocument = errorDocument,
                    },
                };
                PutBucketWebsiteResponse response = await
client.PutBucketWebsiteAsync(putRequest);
```

- Weitere API-Informationen finden Sie unter PutBucketWebsite in der APIAWS SDK for .NET -Referenz für .

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::PutWebsiteConfig(const Aws::String &bucketName,
                                  const Aws::String &indexPage, const Aws::String
&errorPage,
                                  const Aws::Client::ClientConfiguration
&clientConfig) {
   Aws::S3::S3Client client(clientConfig);

   Aws::S3::Model::IndexDocument indexDocument;
   indexDocument.SetSuffix(indexPage);

   Aws::S3::Model::ErrorDocument errorDocument;
   errorDocument.SetKey(errorPage);

   Aws::S3::Model::WebsiteConfiguration websiteConfiguration;
   websiteConfiguration.SetIndexDocument(indexDocument);
   websiteConfiguration.SetErrorDocument(errorDocument);

   Aws::S3::Model::PutBucketWebsiteRequest request;
   request.SetBucket(bucketName);
```

```
    request.SetWebsiteConfiguration(websiteConfiguration);

    Aws::S3::Model::PutBucketWebsiteOutcome outcome =
            client.PutBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutBucketWebsite: "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Success: Set website configuration for bucket '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter PutBucketWebsite in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Der wendet eine statische Website-Konfiguration auf einen Bucket mit dem Namen an`my-bucket`:

```
aws s3api put-bucket-website --bucket my-bucket --website-configuration file://
website.json
```

Die Datei `website.json` ist ein JSON-Dokument im aktuellen Ordner, das Index- und Fehlerseiten für die Website angibt:

```
{
    "IndexDocument": {
        "Suffix": "index.html"
    },
    "ErrorDocument": {
        "Key": "error.html"
    }
```

```
}
```

- API-Details finden Sie unter PutBucketWebsite in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.IndexDocument;
import software.amazon.awssdk.services.s3.model.PutBucketWebsiteRequest;
import software.amazon.awssdk.services.s3.model.WebsiteConfiguration;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class SetWebsiteConfiguration {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <bucketName> [indexdoc]\s

                Where:
                   bucketName   - The Amazon S3 bucket to set the website
 configuration on.\s
                   indexdoc - The index document, ex. 'index.html'
                              If not specified, 'index.html' will be set.
```

```
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String indexDoc = "index.html";
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setWebsiteConfig(s3, bucketName, indexDoc);
        s3.close();
    }

    public static void setWebsiteConfig(S3Client s3, String bucketName, String
 indexDoc) {
        try {
            WebsiteConfiguration websiteConfig = WebsiteConfiguration.builder()

 .indexDocument(IndexDocument.builder().suffix(indexDoc).build())
                    .build();

            PutBucketWebsiteRequest pubWebsiteReq =
 PutBucketWebsiteRequest.builder()
                    .bucket(bucketName)
                    .websiteConfiguration(websiteConfig)
                    .build();

            s3.putBucketWebsite(pubWebsiteReq);
            System.out.println("The call was successful");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter PutBucketWebsite in der APIAWS SDK for Java 2.x -Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Legen Sie die Website-Konfiguration fest.

```
import { PutBucketWebsiteCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// Set up a bucket as a static website.
// The bucket needs to be publicly accessible.
export const main = async () => {
  const command = new PutBucketWebsiteCommand({
    Bucket: "test-bucket",
    WebsiteConfiguration: {
      ErrorDocument: {
        // The object key name to use when a 4XX class error occurs.
        Key: "error.html",
      },
      IndexDocument: {
        // A suffix that is appended to a request that is for a directory.
        Suffix: "index.html",
      },
    },
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
```

```
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter PutBucketWebsite in der APIAWS SDK for JavaScript -Referenz für .

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket website actions.
class BucketWebsiteWrapper
  attr_reader :bucket_website

  # @param bucket_website [Aws::S3::BucketWebsite] A bucket website object
 configured with an existing bucket.
  def initialize(bucket_website)
    @bucket_website = bucket_website
  end

  # Sets a bucket as a static website.
  #
  # @param index_document [String] The name of the index document for the
 website.
  # @param error_document [String] The name of the error document to show for 4XX
 errors.
  # @return [Boolean] True when the bucket is configured as a website; otherwise,
 false.
  def set_website(index_document, error_document)
    @bucket_website.put(
      website_configuration: {
        index_document: { suffix: index_document },
```

```
        error_document: { key: error_document }
      }
    )
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't configure #{@bucket_website.bucket.name} as a website. Here's
 why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  index_document = "index.html"
  error_document = "404.html"

  wrapper = BucketWebsiteWrapper.new(Aws::S3::BucketWebsite.new(bucket_name))
  return unless wrapper.set_website(index_document, error_document)

  puts "Successfully configured bucket #{bucket_name} as a static website."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter PutBucketWebsite in der APIAWS SDK for Ruby -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Beispielansätze für Komponenten- und Integrationstests mit einem - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie Beispiele für bewährte Methoden beim Schreiben von Einheiten- und Integrationstests mit einem AWS -SDK erstellen.

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Cargo.toml für Testbeispiele

```
[package]
name = "testing-examples"
version = "0.1.0"
authors = [
  "John Disanti <jdisanti@amazon.com>",
  "Doug Schwartz <dougsch@amazon.com>",
]
edition = "2021"

# snippet-start:[testing.rust.Cargo.toml]
[dependencies]
async-trait = "0.1.51"
aws-config = { version = "1.0.1", features = ["behavior-version-latest"] }
aws-credential-types = { version = "1.0.1", features = [ "hardcoded-
credentials", ] }
aws-sdk-s3 = { version = "1.4.0" }
aws-smithy-types = { version = "1.0.1" }
aws-smithy-runtime = { version = "1.0.1", features = ["test-util"] }
aws-smithy-runtime-api = { version = "1.0.1", features = ["test-util"] }
aws-types = { version = "1.0.1" }
clap = { version = "~4.4", features = ["derive"] }
http = "0.2.9"
mockall = "0.11.4"
serde_json = "1"
tokio = { version = "1.20.1", features = ["full"] }
tracing-subscriber = { version = "0.3.15", features = ["env-filter"] }
# snippet-end:[testing.rust.Cargo.toml]

[[bin]]
name = "main"
path = "src/main.rs"
```

Beispiel für Modultests mit automock und einem Service-Wrapper

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

// snippet-start:[testing.rust.wrapper]
// snippet-start:[testing.rust.wrapper-uses]
use aws_sdk_s3 as s3;
#[allow(unused_imports)]
use mockall::automock;

use s3::operation::list_objects_v2::{ListObjectsV2Error, ListObjectsV2Output};
// snippet-end:[testing.rust.wrapper-uses]

// snippet-start:[testing.rust.wrapper-which-impl]
#[cfg(test)]
pub use MockS3Impl as S3;
#[cfg(not(test))]
pub use S3Impl as S3;
// snippet-end:[testing.rust.wrapper-which-impl]

// snippet-start:[testing.rust.wrapper-impl]
#[allow(dead_code)]
pub struct S3Impl {
    inner: s3::Client,
}

#[cfg_attr(test, automock)]
impl S3Impl {
    #[allow(dead_code)]
    pub fn new(inner: s3::Client) -> Self {
        Self { inner }
    }

    #[allow(dead_code)]
    pub async fn list_objects(
        &self,
        bucket: &str,
        prefix: &str,
        continuation_token: Option<String>,
    ) -> Result<ListObjectsV2Output, s3::error::SdkError<ListObjectsV2Error>> {
```

```
        self.inner
            .list_objects_v2()
            .bucket(bucket)
            .prefix(prefix)
            .set_continuation_token(continuation_token)
            .send()
            .await
    }
}
// snippet-end:[testing.rust.wrapper-impl]

// snippet-start:[testing.rust.wrapper-func]
#[allow(dead_code)]
pub async fn determine_prefix_file_size(
    // Now we take a reference to our trait object instead of the S3 client
    // s3_list: ListObjectsService,
    s3_list: S3,
    bucket: &str,
    prefix: &str,
) -> Result<usize, s3::Error> {
    let mut next_token: Option<String> = None;
    let mut total_size_bytes = 0;
    loop {
        let result = s3_list
            .list_objects(bucket, prefix, next_token.take())
            .await?;

        // Add up the file sizes we got back
        for object in result.contents() {
            total_size_bytes += object.size().unwrap_or(0) as usize;
        }

        // Handle pagination, and break the loop if there are no more pages
        next_token = result.next_continuation_token.clone();
        if next_token.is_none() {
            break;
        }
    }
    Ok(total_size_bytes)
}
// snippet-end:[testing.rust.wrapper-func]
// snippet-end:[testing.rust.wrapper]

// snippet-start:[testing.rust.wrapper-test-mod]
```

```
#[cfg(test)]
mod test {
    // snippet-start:[testing.rust.wrapper-tests]
    use super::*;
    use mockall::predicate::eq;

    // snippet-start:[testing.rust.wrapper-test-single]
    #[tokio::test]
    async fn test_single_page() {
        let mut mock = MockS3Impl::default();
        mock.expect_list_objects()
            .with(eq("test-bucket"), eq("test-prefix"), eq(None))
            .return_once(|_, _, _| {
                Ok(ListObjectsV2Output::builder()
                    .set_contents(Some(vec![
                        // Mock content for ListObjectsV2 response
                        s3::types::Object::builder().size(5).build(),
                        s3::types::Object::builder().size(2).build(),
                    ]))
                    .build())
            });

        // Run the code we want to test with it
        let size = determine_prefix_file_size(mock, "test-bucket", "test-prefix")
            .await
            .unwrap();

        // Verify we got the correct total size back
        assert_eq!(7, size);
    }
    // snippet-end:[testing.rust.wrapper-test-single]

    // snippet-start:[testing.rust.wrapper-test-multiple]
    #[tokio::test]
    async fn test_multiple_pages() {
        // Create the Mock instance with two pages of objects now
        let mut mock = MockS3Impl::default();
        mock.expect_list_objects()
            .with(eq("test-bucket"), eq("test-prefix"), eq(None))
            .return_once(|_, _, _| {
                Ok(ListObjectsV2Output::builder()
                    .set_contents(Some(vec![
                        // Mock content for ListObjectsV2 response
                        s3::types::Object::builder().size(5).build(),
```

```
                        s3::types::Object::builder().size(2).build(),
                    ]))
                    .set_next_continuation_token(Some("next".to_string()))
                    .build())
            });
        mock.expect_list_objects()
            .with(
                eq("test-bucket"),
                eq("test-prefix"),
                eq(Some("next".to_string())),
            )
            .return_once(|_, _, _| {
                Ok(ListObjectsV2Output::builder()
                    .set_contents(Some(vec![
                        // Mock content for ListObjectsV2 response
                        s3::types::Object::builder().size(3).build(),
                        s3::types::Object::builder().size(9).build(),
                    ]))
                    .build())
            });

        // Run the code we want to test with it
        let size = determine_prefix_file_size(mock, "test-bucket", "test-prefix")
            .await
            .unwrap();

        assert_eq!(19, size);
    }
    // snippet-end:[testing.rust.wrapper-test-multiple]
    // snippet-end:[testing.rust.wrapper-tests]
}
// snippet-end:[testing.rust.wrapper-test-mod]
```

Beispiel für Integrationstests mit StaticReplayClient.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

// snippet-start:[testing.rust.replay-uses]
use aws_sdk_s3 as s3;
// snippet-end:[testing.rust.replay-uses]
```

```
#[allow(dead_code)]
// snippet-start:[testing.rust.replay]
pub async fn determine_prefix_file_size(
    // Now we take a reference to our trait object instead of the S3 client
    // s3_list: ListObjectsService,
    s3: s3::Client,
    bucket: &str,
    prefix: &str,
) -> Result<usize, s3::Error> {
    let mut next_token: Option<String> = None;
    let mut total_size_bytes = 0;
    loop {
        let result = s3
            .list_objects_v2()
            .prefix(prefix)
            .bucket(bucket)
            .set_continuation_token(next_token.take())
            .send()
            .await?;

        // Add up the file sizes we got back
        for object in result.contents() {
            total_size_bytes += object.size().unwrap_or(0) as usize;
        }

        // Handle pagination, and break the loop if there are no more pages
        next_token = result.next_continuation_token.clone();
        if next_token.is_none() {
            break;
        }
    }
    Ok(total_size_bytes)
}
// snippet-end:[testing.rust.replay]

#[allow(dead_code)]
// snippet-start:[testing.rust.replay-tests]
// snippet-start:[testing.rust.replay-make-credentials]
fn make_s3_test_credentials() -> s3::config::Credentials {
    s3::config::Credentials::new(
        "ATESTCLIENT",
        "astestsecretkey",
        Some("atestsessiontoken".to_string()),
        None,
```

```
        "",
    )
}
// snippet-end:[testing.rust.replay-make-credentials]

// snippet-start:[testing.rust.replay-test-module]
#[cfg(test)]
mod test {
    // snippet-start:[testing.rust.replay-test-single]
    use super::*;
    use aws_config::BehaviorVersion;
    use aws_sdk_s3 as s3;
    use aws_smithy_runtime::client::http::test_util::{ReplayEvent,
 StaticReplayClient};
    use aws_smithy_types::body::SdkBody;

    #[tokio::test]
    async fn test_single_page() {
        let page_1 = ReplayEvent::new(
                http::Request::builder()
                    .method("GET")
                    .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix")
                    .body(SdkBody::empty())
                    .unwrap(),
                http::Response::builder()
                    .status(200)
                    .body(SdkBody::from(include_str!("./testing/
response_1.xml")))
                    .unwrap(),
            );
        let replay_client = StaticReplayClient::new(vec![page_1]);
        let client: s3::Client = s3::Client::from_conf(
            s3::Config::builder()
                .behavior_version(BehaviorVersion::latest())
                .credentials_provider(make_s3_test_credentials())
                .region(s3::config::Region::new("us-east-1"))
                .http_client(replay_client.clone())
                .build(),
        );

        // Run the code we want to test with it
        let size = determine_prefix_file_size(client, "test-bucket", "test-
prefix")
```

```
            .await
            .unwrap();

        // Verify we got the correct total size back
        assert_eq!(7, size);
        replay_client.assert_requests_match(&[]);
    }
    // snippet-end:[testing.rust.replay-test-single]

    // snippet-start:[testing.rust.replay-test-multiple]
    #[tokio::test]
    async fn test_multiple_pages() {
        // snippet-start:[testing.rust.replay-create-replay]
        let page_1 = ReplayEvent::new(
                http::Request::builder()
                    .method("GET")
                    .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix")
                    .body(SdkBody::empty())
                    .unwrap(),
                http::Response::builder()
                    .status(200)
                    .body(SdkBody::from(include_str!("./testing/
response_multi_1.xml")))
                    .unwrap(),
            );
        let page_2 = ReplayEvent::new(
                http::Request::builder()
                    .method("GET")
                    .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix&continuation-token=next")
                    .body(SdkBody::empty())
                    .unwrap(),
                http::Response::builder()
                    .status(200)
                    .body(SdkBody::from(include_str!("./testing/
response_multi_2.xml")))
                    .unwrap(),
            );
        let replay_client = StaticReplayClient::new(vec![page_1, page_2]);
        // snippet-end:[testing.rust.replay-create-replay]
        // snippet-start:[testing.rust.replay-create-client]
        let client: s3::Client = s3::Client::from_conf(
            s3::Config::builder()
```

```
                .behavior_version(BehaviorVersion::latest())
                .credentials_provider(make_s3_test_credentials())
                .region(s3::config::Region::new("us-east-1"))
                .http_client(replay_client.clone())
                .build(),
        );
        // snippet-end:[testing.rust.replay-create-client]

        // Run the code we want to test with it
        // snippet-start:[testing.rust.replay-test-and-verify]
        let size = determine_prefix_file_size(client, "test-bucket", "test-
prefix")
            .await
            .unwrap();

        assert_eq!(19, size);

        replay_client.assert_requests_match(&[]);
        // snippet-end:[testing.rust.replay-test-and-verify]
    }
    // snippet-end:[testing.rust.replay-test-multiple]
}
// snippet-end:[testing.rust.replay-tests]
// snippet-end:[testing.rust.replay-test-module]
```

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Hochladen eines einzelnen Teils eines mehrteiligen Uploads mithilfe eines - AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie einen einzelnen Teil eines mehrteiligen Uploads hochladen.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Durchführen eines mehrteiligen Uploads
- Verwenden der Prüfsummen

CLI

AWS CLI

Der folgende Befehl lädt den ersten Teil eines mehrteiligen Uploads hoch, der mit dem `create-multipart-upload` Befehl initiiert wurde:

```
aws s3api upload-part --bucket my-bucket --key 'multipart/01' --part-number 1 --
body part01 --upload-id
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3
```

Die `body` Option verwendet den Namen oder Pfad einer lokalen Datei zum Hochladen (verwenden Sie nicht das Präfix file://). Die minimale Teilegröße beträgt 5 MB. Die Upload-ID wird von zurückgegeben `create-multipart-upload` und kann auch mit abgerufen werden`list-multipart-uploads`. Bucket und Schlüssel werden angegeben, wenn Sie den mehrteiligen Upload erstellen.

Ausgabe:

```
{
    "ETag": "\"e868e0f4719e394144ef36531ee6824c\""
}
```

Speichern Sie den ETag-Wert jedes Teils für später. Sie sind erforderlich, um den mehrteiligen Upload abzuschließen.

- API-Details finden Sie unter UploadPart in der AWS CLI -Befehlsreferenz.

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        let upload_part_res = client
```

```
            .upload_part()
            .key(&key)
            .bucket(&bucket_name)
            .upload_id(upload_id)
            .body(stream)
            .part_number(part_number)
            .send()
            .await?;
        upload_parts.push(
            CompletedPart::builder()
                .e_tag(upload_part_res.e_tag.unwrap_or_default())
                .part_number(part_number)
                .build(),
        );

   let completed_multipart_upload: CompletedMultipartUpload =
CompletedMultipartUpload::builder()
        .set_parts(Some(upload_parts))
        .build();
```

- Weitere API-Informationen finden Sie unter UploadPart in der API-AWS Referenz zum -SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Hochladen eines Objekts in einen Amazon S3-Bucket mithilfe eines - AWS SDK

In den folgenden Codebeispielen wird veranschaulicht, wie Sie ein Objekt in einen S3 Bucket hochladen.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Buckets und Objekten

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        /// <summary>
        /// Shows how to upload a file from the local computer to an Amazon S3
        /// bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The Amazon S3 bucket to which the object
        /// will be uploaded.</param>
        /// <param name="objectName">The object to upload.</param>
        /// <param name="filePath">The path, including file name, of the object
        /// on the local computer to upload.</param>
        /// <returns>A boolean value indicating the success or failure of the
        /// upload procedure.</returns>
        public static async Task<bool> UploadFileAsync(
            IAmazonS3 client,
            string bucketName,
            string objectName,
            string filePath)
        {
            var request = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = objectName,
                FilePath = filePath,
            };

            var response = await client.PutObjectAsync(request);
            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"Successfully uploaded {objectName} to
{bucketName}.");
                return true;
```

```
            }
            else
            {
                Console.WriteLine($"Could not upload {objectName} to
{bucketName}.");
                return false;
            }
        }
```

Laden Sie ein Objekt mit serverseitiger Verschlüsselung hoch.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to upload an object to an Amazon Simple Storage
    /// Service (Amazon S3) bucket with server-side encryption enabled.
    /// </summary>
    public class ServerSideEncryption
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "samplefile.txt";

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the Amazon S3 client object's constructor.
            // For example: RegionEndpoint.USWest2.
            IAmazonS3 client = new AmazonS3Client();

            await WritingAnObjectAsync(client, bucketName, keyName);
        }

        /// <summary>
        /// Upload a sample object include a setting for encryption.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
        /// to upload a file and apply server-side encryption.</param>
```

```
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
        /// encrypted object will reside.</param>
        /// <param name="keyName">The name for the object that you want to
        /// create in the supplied bucket.</param>
        public static async Task WritingAnObjectAsync(IAmazonS3 client, string
bucketName, string keyName)
        {
            try
            {
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    ContentBody = "sample text",
                    ServerSideEncryptionMethod =
ServerSideEncryptionMethod.AES256,
                };

                var putResponse = await client.PutObjectAsync(putRequest);

                // Determine the encryption state of an object.
                GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                };
                GetObjectMetadataResponse response = await
client.GetObjectMetadataAsync(metadataRequest);
                ServerSideEncryptionMethod objectEncryption =
response.ServerSideEncryptionMethod;

                Console.WriteLine($"Encryption method used: {0}",
objectEncryption.ToString());
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: '{ex.Message}' when writing an
object");
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter PutObject in der APIAWS SDK for .NET - Referenz für .

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file to.
#       $2 - The path and file name of the local file to copy to the bucket.
#       $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function copy_file_to_bucket() {
  local response bucket_name source_file destination_file_name
  bucket_name=$1
```

```
  source_file=$2
  destination_file_name=$3

  response=$(aws s3api put-object \
    --bucket "$bucket_name" \
    --body "$source_file" \
    --key "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}
```

- API-Details finden Sie unter PutObject in der AWS CLI -Befehlsreferenz.

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
bool AwsDoc::S3::PutObject(const Aws::String &bucketName,
                           const Aws::String &fileName,
                           const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucketName);
    //We are using the name of the file as the key for the object in the bucket.
    //However, this is just a string and can be set according to your retrieval
 needs.
    request.SetKey(fileName);

    std::shared_ptr<Aws::IOStream> inputData =
```

```
            Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                          fileName.c_str(),
                                          std::ios_base::in |
 std::ios_base::binary);

    if (!*inputData) {
        std::cerr << "Error unable to read file " << fileName << std::endl;
        return false;
    }

    request.SetBody(inputData);

    Aws::S3::Model::PutObjectOutcome outcome =
            s3_client.PutObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutObject: " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Added object '" << fileName << "' to bucket '"
                  << bucketName << "'.";
    }

    return outcome.IsSuccess();
}
```

- Weitere API-Informationen finden Sie unter PutObject in der APIAWS SDK for C++ - Referenz für .

CLI

AWS CLI

Im folgenden Beispiel wird der `put-object` Befehl verwendet, um ein Objekt in Amazon S3 hochzuladen:

```
aws s3api put-object --bucket text-content --key dir-1/my_images.tar.bz2 --body
 my_images.tar.bz2
```

Das folgende Beispiel zeigt einen Upload einer Videodatei (Die Videodatei wird mithilfe der Windows-Dateisystemsyntax angegeben.):

```
aws s3api put-object --bucket text-content --key dir-1/big-video-file.mp4 --body
 e:\media\videos\f-sharp-3-data-services.mp4
```

Weitere Informationen zum Hochladen von Objekten finden Sie unter Hochladen von Objekten im Amazon S3-Entwicklerhandbuch.

- API-Details finden Sie unter PutObject in der AWS CLI -Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// UploadFile reads from a file and puts the data into an object in a bucket.
func (basics BucketBasics) UploadFile(bucketName string, objectKey string,
 fileName string) error {
 file, err := os.Open(fileName)
 if err != nil {
  log.Printf("Couldn't open file %v to upload. Here's why: %v\n", fileName, err)
 } else {
```

```
  defer file.Close()
  _, err = basics.S3Client.PutObject(context.TODO(), &s3.PutObjectInput{
   Bucket: aws.String(bucketName),
   Key:    aws.String(objectKey),
   Body:   file,
  })
  if err != nil {
   log.Printf("Couldn't upload file %v to %v:%v. Here's why: %v\n",
    fileName, bucketName, objectKey, err)
  }
 }
 return err
}
```

- Weitere API-Informationen finden Sie unter PutObject in der APIAWS SDK for Go -Referenz für .

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Laden Sie eine Datei mit einem S3Client in einen Bucket hoch.

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
```

```
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class PutObject {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <bucketName> <objectKey> <objectPath>\s

                Where:
                  bucketName - The Amazon S3 bucket to upload an object into.
                  objectKey - The object to upload (for example, book.pdf).
                  objectPath - The path where the file is located (for example,
 C:/AWS/book2.pdf).\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String objectKey = args[1];
        String objectPath = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        putS3Object(s3, bucketName, objectKey, objectPath);
        s3.close();
    }

    // This example uses RequestBody.fromFile to avoid loading the whole file
 into
    // memory.
    public static void putS3Object(S3Client s3, String bucketName, String
 objectKey, String objectPath) {
```

```
        try {
            Map<String, String> metadata = new HashMap<>();
            metadata.put("x-amz-meta-myVal", "test");
            PutObjectRequest putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .metadata(metadata)
                    .build();

            s3.putObject(putOb, RequestBody.fromFile(new File(objectPath)));
            System.out.println("Successfully placed " + objectKey + " into bucket
 " + bucketName);

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

Verwenden Sie einen S3TransferManager, um eine Datei in einen Bucket hochzuladen. Sehen Sie sich die vollständige Datei an und testen Sie sie.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedFileUpload;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;
import software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener;
import java.net.URI;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Paths;
import java.util.UUID;

    public String uploadFile(S3TransferManager transferManager, String
 bucketName,
                             String key, URI filePathURI) {
        UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
            .putObjectRequest(b -> b.bucket(bucketName).key(key))
            .addTransferListener(LoggingTransferListener.create())
```

```
            .source(Paths.get(filePathURI))
            .build();

        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);

        CompletedFileUpload uploadResult = fileUpload.completionFuture().join();
        return uploadResult.response().eTag();
    }
```

Laden Sie ein Objekt mit einem S3Client in einen Bucket hoch und legen Sie Tags fest.

```
    public static void putS3ObjectTags(S3Client s3, String bucketName, String
objectKey, String objectPath) {
        try {
            Tag tag1 = Tag.builder()
                    .key("Tag 1")
                    .value("This is tag 1")
                    .build();

            Tag tag2 = Tag.builder()
                    .key("Tag 2")
                    .value("This is tag 2")
                    .build();

            List<Tag> tags = new ArrayList<>();
            tags.add(tag1);
            tags.add(tag2);

            Tagging allTags = Tagging.builder()
                    .tagSet(tags)
                    .build();

            PutObjectRequest putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .tagging(allTags)
                    .build();

            s3.putObject(putOb,
RequestBody.fromBytes(getObjectFile(objectPath)));

        } catch (S3Exception e) {
```

```
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void updateObjectTags(S3Client s3, String bucketName, String
objectKey) {
        try {
            GetObjectTaggingRequest taggingRequest =
GetObjectTaggingRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .build();

            GetObjectTaggingResponse getTaggingRes =
s3.getObjectTagging(taggingRequest);
            List<Tag> obTags = getTaggingRes.tagSet();
            for (Tag sinTag : obTags) {
                System.out.println("The tag key is: " + sinTag.key());
                System.out.println("The tag value is: " + sinTag.value());
            }

            // Replace the object's tags with two new tags.
            Tag tag3 = Tag.builder()
                    .key("Tag 3")
                    .value("This is tag 3")
                    .build();

            Tag tag4 = Tag.builder()
                    .key("Tag 4")
                    .value("This is tag 4")
                    .build();

            List<Tag> tags = new ArrayList<>();
            tags.add(tag3);
            tags.add(tag4);

            Tagging updatedTags = Tagging.builder()
                    .tagSet(tags)
                    .build();

            PutObjectTaggingRequest taggingRequest1 =
PutObjectTaggingRequest.builder()
                    .bucket(bucketName)
```

```
                    .key(objectKey)
                    .tagging(updatedTags)
                    .build();

            s3.putObjectTagging(taggingRequest1);
            GetObjectTaggingResponse getTaggingRes2 =
s3.getObjectTagging(taggingRequest);
            List<Tag> modTags = getTaggingRes2.tagSet();
            for (Tag sinTag : modTags) {
                System.out.println("The tag key is: " + sinTag.key());
                System.out.println("The tag value is: " + sinTag.value());
            }

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    // Return a byte array.
    private static byte[] getObjectFile(String filePath) {
        FileInputStream fileInputStream = null;
        byte[] bytesArray = null;

        try {
            File file = new File(filePath);
            bytesArray = new byte[(int) file.length()];
            fileInputStream = new FileInputStream(file);
            fileInputStream.read(bytesArray);

        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (fileInputStream != null) {
                try {
                    fileInputStream.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        return bytesArray;
    }
```

```
}
```

Laden Sie ein Objekt mit einem S3Client in einen Bucket hoch und legen Sie Metadaten fest.

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class PutObjectMetadata {
    public static void main(String[] args) {
        final String USAGE = """

                Usage:
                  <bucketName> <objectKey> <objectPath>\s

                Where:
                  bucketName - The Amazon S3 bucket to upload an object into.
                  objectKey - The object to upload (for example, book.pdf).
                  objectPath - The path where the file is located (for example,
 C:/AWS/book2.pdf).\s
                """;

        if (args.length != 3) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String bucketName = args[0];
```

```
        String objectKey = args[1];
        String objectPath = args[2];
        System.out.println("Putting object " + objectKey + " into bucket " +
 bucketName);
        System.out.println("  in bucket: " + bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        putS3Object(s3, bucketName, objectKey, objectPath);
        s3.close();
    }

    // This example uses RequestBody.fromFile to avoid loading the whole file
 into
    // memory.
    public static void putS3Object(S3Client s3, String bucketName, String
 objectKey, String objectPath) {
        try {
            Map<String, String> metadata = new HashMap<>();
            metadata.put("author", "Mary Doe");
            metadata.put("version", "1.0.0.0");

            PutObjectRequest putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .metadata(metadata)
                    .build();

            s3.putObject(putOb, RequestBody.fromFile(new File(objectPath)));
            System.out.println("Successfully placed " + objectKey + " into bucket
 " + bucketName);

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

Laden Sie ein Objekt mit einem S3Client in einen Bucket hoch und legen Sie einen Wert für die Objektaufbewahrung fest.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRetentionRequest;
import software.amazon.awssdk.services.s3.model.ObjectLockRetention;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.time.Instant;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.ZoneOffset;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class PutObjectRetention {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <key> <bucketName>\s

                Where:
                    key - The name of the object (for example, book.pdf).\s
                    bucketName - The Amazon S3 bucket name that contains the
 object (for example, bucket1).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String key = args[0];
        String bucketName = args[1];
```

```
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setRentionPeriod(s3, key, bucketName);
        s3.close();
    }

    public static void setRentionPeriod(S3Client s3, String key, String bucket) {
        try {
            LocalDate localDate = LocalDate.parse("2020-07-17");
            LocalDateTime localDateTime = localDate.atStartOfDay();
            Instant instant = localDateTime.toInstant(ZoneOffset.UTC);

            ObjectLockRetention lockRetention = ObjectLockRetention.builder()
                    .mode("COMPLIANCE")
                    .retainUntilDate(instant)
                    .build();

            PutObjectRetentionRequest retentionRequest =
 PutObjectRetentionRequest.builder()
                    .bucket(bucket)
                    .key(key)
                    .bypassGovernanceRetention(true)
                    .retention(lockRetention)
                    .build();

            // To set Retention on an object, the Amazon S3 bucket must support
 object
            // locking, otherwise an exception is thrown.
            s3.putObjectRetention(retentionRequest);
            System.out.print("An object retention configuration was successfully
 placed on the object");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie unter PutObject in der APIAWS SDK for Java 2.x - Referenz für .

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Laden Sie das Objekt hoch.

```
import { PutObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new PutObjectCommand({
    Bucket: "test-bucket",
    Key: "hello-s3.txt",
    Body: "Hello S3!",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.
- Weitere API-Informationen finden Sie unter PutObject in der APIAWS SDK for JavaScript - Referenz für .

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun putS3Object(bucketName: String, objectKey: String, objectPath:
 String) {
    val metadataVal = mutableMapOf<String, String>()
    metadataVal["myVal"] = "test"

    val request = PutObjectRequest {
        bucket = bucketName
        key = objectKey
        metadata = metadataVal
        body = File(objectPath).asByteStream()
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        val response = s3.putObject(request)
        println("Tag information is ${response.eTag}")
    }
}
```

- Weitere API-Informationen finden Sie unter PutObject in der AWS API-Referenz zum -SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Laden Sie ein Objekt in einen Bucket hoch.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        $fileName = __DIR__ . "/local-file-" . uniqid();
        try {
            $this->s3client->putObject([
                'Bucket' => $this->bucketName,
                'Key' => $fileName,
                'SourceFile' => __DIR__ . '/testfile.txt'
            ]);
            echo "Uploaded $fileName to $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to upload $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with file upload before continuing.");
        }
```

- Weitere API-Informationen finden Sie unter PutObject in der APIAWS SDK for PHP - Referenz für .

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
```

```
        self.object = s3_object
        self.key = self.object.key


    def put(self, data):
        """
        Upload data to the object.

        :param data: The data to upload. This can either be bytes or a string.
When this
                     argument is a string, it is interpreted as a file name,
which is
                     opened in read bytes mode.
        """
        put_data = data
        if isinstance(data, str):
            try:
                put_data = open(data, "rb")
            except IOError:
                logger.exception("Expected file name or binary data, got '%s'.",
data)
                raise

        try:
            self.object.put(Body=put_data)
            self.object.wait_until_exists()
            logger.info(
                "Put object '%s' to bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't put object '%s' to bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
            raise
        finally:
            if getattr(put_data, "close", None):
                put_data.close()
```

- Weitere API-Informationen finden Sie unter PutObject in der AWS API-Referenz zum -SDK für Python (Boto3).

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Laden Sie eine Datei mit einem verwalteten Uploader (Object.upload_file) hoch.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectUploadFileWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Uploads a file to an Amazon S3 object by using a managed uploader.
  #
  # @param file_path [String] The path to the file to upload.
  # @return [Boolean] True when the file is uploaded; otherwise false.
  def upload_file(file_path)
    @object.upload_file(file_path)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't upload file #{file_path} to #{@object.key}. Here's why:
 #{e.message}"
    false
  end
end

# Example usage:
def run_demo
```

```
  bucket_name = "doc-example-bucket"
  object_key = "my-uploaded-file"
  file_path = "object_upload_file.rb"

  wrapper = ObjectUploadFileWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  return unless wrapper.upload_file(file_path)

  puts "File #{file_path} successfully uploaded to #{bucket_name}:#{object_key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Laden Sie eine Datei mithilfe von Object.put hoch.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectPutWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object(source_file_path)
    File.open(source_file_path, "rb") do |file|
      @object.put(body: file)
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put #{source_file_path} to #{object.key}. Here's why:
 #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object-key"
```

```
  file_path = "my-local-file.txt"

  wrapper = ObjectPutWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  success = wrapper.put_object(file_path)
  return unless success

  puts "Put file #{file_path} into #{object_key} in #{bucket_name}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Laden Sie eine Datei mithilfe von Object.put hoch und fügen Sie eine serverseitige Verschlüsselung hinzu.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectPutSseWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object_encrypted(object_content, encryption)
    @object.put(body: object_content, server_side_encryption: encryption)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put your content to #{object.key}. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-encrypted-content"
  object_content = "This is my super-secret content."
  encryption = "AES256"
```

```
  wrapper = ObjectPutSseWrapper.new(Aws::S3::Object.new(bucket_name,
 object_content))
  return unless wrapper.put_object_encrypted(object_content, encryption)

  puts "Put your content into #{bucket_name}:#{object_key} and encrypted it with
 #{encryption}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Weitere API-Informationen finden Sie unter PutObject in der APIAWS SDK for Ruby - Referenz für .

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
pub async fn upload_object(
    client: &Client,
    bucket_name: &str,
    file_name: &str,
    key: &str,
) -> Result<PutObjectOutput, SdkError<PutObjectError>> {
    let body = ByteStream::from_path(Path::new(file_name)).await;
    client
        .put_object()
        .bucket(bucket_name)
        .key(key)
        .body(body.unwrap())
        .send()
        .await
}
```

- Weitere API-Informationen finden Sie unter PutObject in der API-AWS Referenz zum -SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
"Get contents of file from application server."
DATA lv_body TYPE xstring.
OPEN DATASET iv_file_name FOR INPUT IN BINARY MODE.
READ DATASET iv_file_name INTO lv_body.
CLOSE DATASET iv_file_name.

"Upload/put an object to an S3 bucket."
TRY.
    lo_s3->putobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_file_name
        iv_body = lv_body
    ).
    MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
```

- Weitere API-Informationen finden Sie unter PutObject in der AWS API-Referenz zum -SDK für SAP ABAP.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Eine Datei aus dem lokalen Speicher in einen Bucket hochladen

```
   public func uploadFile(bucket: String, key: String, file: String) async
throws {
       let fileUrl = URL(fileURLWithPath: file)
       let fileData = try Data(contentsOf: fileUrl)
       let dataStream = ByteStream.from(data: fileData)

       let input = PutObjectInput(
           body: dataStream,
           bucket: bucket,
           key: key
       )
       _ = try await client.putObject(input: input)
   }
```

Den Inhalt eines Swift-Data-Objekts in einen Bucket hochladen

```
   public func createFile(bucket: String, key: String, withData data: Data)
async throws {
       let dataStream = ByteStream.from(data: data)

       let input = PutObjectInput(
           body: dataStream,
```

```
            bucket: bucket,
            key: key
        )
        _ = try await client.putObject(input: input)
    }
```

- Weitere API-Informationen finden Sie unter PutObject in der AWS API-Referenz zum -SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Rekursives Hochladen eines lokalen Verzeichnisses in einen Amazon Simple Storage Service (Amazon S3)-Bucket

Das folgende Codebeispiel zeigt, wie Sie ein lokales Verzeichnis rekursiv in einen Amazon Simple Storage Service (Amazon S3)-Bucket hochladen.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Verwenden Sie einen S3TransferManager, um ein lokales Verzeichnis hochzuladen. Sehen Sie sich die vollständige Datei an und testen Sie sie.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedDirectoryUpload;
import software.amazon.awssdk.transfer.s3.model.DirectoryUpload;
```

```
import software.amazon.awssdk.transfer.s3.model.UploadDirectoryRequest;

import java.net.URI;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Paths;
import java.util.UUID;

    public Integer uploadDirectory(S3TransferManager transferManager,
            URI sourceDirectory, String bucketName) {
        DirectoryUpload directoryUpload =
 transferManager.uploadDirectory(UploadDirectoryRequest.builder()
                .source(Paths.get(sourceDirectory))
                .bucket(bucketName)
                .build());

        CompletedDirectoryUpload completedDirectoryUpload =
 directoryUpload.completionFuture().join();
        completedDirectoryUpload.failedTransfers()
                .forEach(fail -> logger.warn("Object [{}] failed to transfer",
 fail.toString()));
        return completedDirectoryUpload.failedTransfers().size();
    }
```

- Weitere API-Informationen finden Sie unter UploadDirectory in der APIAWS SDK for Java 2.x -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden von SQL mit Amazon S3 Select zum Abrufen einer Teilmenge von Daten mithilfe eines AWS SDK

Die folgenden Codebeispiele zeigen, wie Sie eine Teilmenge von Daten mit SQL abrufen.

CLI

AWS CLI

So filtern Sie den Inhalt eines Amazon S3-Objekts basierend auf einer SQL-Anweisung

Das folgende `select-object-content` Beispiel filtert das Objekt `my-data-file.csv` mit der angegebenen SQL-Anweisung und sendet eine Ausgabe an eine Datei.

```
aws s3api select-object-content \
    --bucket my-bucket \
    --key my-data-file.csv \
    --expression "select * from s3object limit 100" \
    --expression-type 'SQL' \
    --input-serialization '{"CSV": {}, "CompressionType": "NONE"}' \
    --output-serialization '{"CSV": {}}' "output.csv"
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- API-Details finden Sie unter SelectObjectContent in der AWS CLI -Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Das folgende Beispiel zeigt eine Abfrage mit einem JSON-Objekt. Das vollständige Beispiel zeigt auch die Verwendung eines CSV-Objekts.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.async.BlockingInputStreamAsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.model.CSVInput;
import software.amazon.awssdk.services.s3.model.CSVOutput;
import software.amazon.awssdk.services.s3.model.CompressionType;
import software.amazon.awssdk.services.s3.model.ExpressionType;
import software.amazon.awssdk.services.s3.model.FileHeaderInfo;
import software.amazon.awssdk.services.s3.model.InputSerialization;
import software.amazon.awssdk.services.s3.model.JSONInput;
```

```
import software.amazon.awssdk.services.s3.model.JSONOutput;
import software.amazon.awssdk.services.s3.model.JSONType;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.services.s3.model.OutputSerialization;
import software.amazon.awssdk.services.s3.model.Progress;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.SelectObjectContentRequest;
import
 software.amazon.awssdk.services.s3.model.SelectObjectContentResponseHandler;
import software.amazon.awssdk.services.s3.model.Stats;

import java.io.IOException;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;

public class SelectObjectContentExample {
    static final Logger logger =
 LoggerFactory.getLogger(SelectObjectContentExample.class);
    static final String BUCKET_NAME = "select-object-content-" +
 UUID.randomUUID();
    static final S3AsyncClient s3AsyncClient = S3AsyncClient.create();
    static String FILE_CSV = "csv";
    static String FILE_JSON = "json";
    static String URL_CSV = "https://raw.githubusercontent.com/mledoze/countries/
master/dist/countries.csv";
    static String URL_JSON = "https://raw.githubusercontent.com/mledoze/
countries/master/dist/countries.json";

    public static void main(String[] args) {
        SelectObjectContentExample selectObjectContentExample = new
 SelectObjectContentExample();
        try {
            SelectObjectContentExample.setUp();
            selectObjectContentExample.runSelectObjectContentMethodForJSON();
            selectObjectContentExample.runSelectObjectContentMethodForCSV();
        } catch (SdkException e) {
            logger.error(e.getMessage(), e);
            System.exit(1);
        } finally {
            SelectObjectContentExample.tearDown();
        }
```

```
    }

    EventStreamInfo runSelectObjectContentMethodForJSON() {
        // Set up request parameters.
        final String queryExpression = "select * from s3object[*][*] c where
c.area < 350000";
        final String fileType = FILE_JSON;

        InputSerialization inputSerialization = InputSerialization.builder()
                .json(JSONInput.builder().type(JSONType.DOCUMENT).build())
                .compressionType(CompressionType.NONE)
                .build();

        OutputSerialization outputSerialization = OutputSerialization.builder()
                .json(JSONOutput.builder().recordDelimiter(null).build())
                .build();

        // Build the SelectObjectContentRequest.
        SelectObjectContentRequest select = SelectObjectContentRequest.builder()
                .bucket(BUCKET_NAME)
                .key(FILE_JSON)
                .expression(queryExpression)
                .expressionType(ExpressionType.SQL)
                .inputSerialization(inputSerialization)
                .outputSerialization(outputSerialization)
                .build();

        EventStreamInfo eventStreamInfo = new EventStreamInfo();
        // Call the selectObjectContent method with the request and a response
handler.
        // Supply an EventStreamInfo object to the response handler to gather
records and information from the response.
        s3AsyncClient.selectObjectContent(select,
buildResponseHandler(eventStreamInfo)).join();

        // Log out information gathered while processing the response stream.
        long recordCount = eventStreamInfo.getRecords().stream().mapToInt(record
->
                record.split("\n").length
        ).sum();
        logger.info("Total records {}: {}", fileType, recordCount);
        logger.info("Visitor onRecords for fileType {} called {} times",
fileType, eventStreamInfo.getCountOnRecordsCalled());
```

```
        logger.info("Visitor onStats for fileType {}, {}", fileType,
 eventStreamInfo.getStats());
        logger.info("Visitor onContinuations for fileType {}, {}", fileType,
 eventStreamInfo.getCountContinuationEvents());
        return eventStreamInfo;
    }

    static SelectObjectContentResponseHandler
 buildResponseHandler(EventStreamInfo eventStreamInfo) {
        // Use a Visitor to process the response stream. This visitor logs
 information and gathers details while processing.
        final SelectObjectContentResponseHandler.Visitor visitor =
 SelectObjectContentResponseHandler.Visitor.builder()
                .onRecords(r -> {
                    logger.info("Record event received.");
                    eventStreamInfo.addRecord(r.payload().asUtf8String());
                    eventStreamInfo.incrementOnRecordsCalled();
                })
                .onCont(ce -> {
                    logger.info("Continuation event received.");
                    eventStreamInfo.incrementContinuationEvents();
                })
                .onProgress(pe -> {
                    Progress progress = pe.details();
                    logger.info("Progress event received:\n bytesScanned:
{}\nbytesProcessed: {}\nbytesReturned:{}",
                            progress.bytesScanned(),
                            progress.bytesProcessed(),
                            progress.bytesReturned());
                })
                .onEnd(ee -> logger.info("End event received."))
                .onStats(se -> {
                    logger.info("Stats event received.");
                    eventStreamInfo.addStats(se.details());
                })
                .build();

        // Build the SelectObjectContentResponseHandler with the visitor that
 processes the stream.
        return SelectObjectContentResponseHandler.builder()
                .subscriber(visitor).build();
    }
```

```
   // The EventStreamInfo class is used to store information gathered while
processing the response stream.
   static class EventStreamInfo {
       private final List<String> records = new ArrayList<>();
       private Integer countOnRecordsCalled = 0;
       private Integer countContinuationEvents = 0;
       private Stats stats;

       void incrementOnRecordsCalled() {
           countOnRecordsCalled++;
       }

       void incrementContinuationEvents() {
           countContinuationEvents++;
       }

       void addRecord(String record) {
           records.add(record);
       }

       void addStats(Stats stats) {
           this.stats = stats;
       }

       public List<String> getRecords() {
           return records;
       }

       public Integer getCountOnRecordsCalled() {
           return countOnRecordsCalled;
       }

       public Integer getCountContinuationEvents() {
           return countContinuationEvents;
       }

       public Stats getStats() {
           return stats;
       }
   }
```

- Weitere API-Informationen finden Sie unter SelectObjectContent in der APIAWS SDK for Java 2.x -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Szenarien für Amazon S3 unter Verwendung von AWS SDKs

Die folgenden Codebeispiele veranschaulichen, wie Sie gängige Szenarien in Amazon S3 mit - AWS SDKs implementieren. Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben ausführen können, indem Sie mehrere Funktionen in Amazon S3 aufrufen. Jedes Szenario enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Beispiele

- Erstellen einer vorsignierten URL für Amazon S3 mithilfe eines - AWS SDK
- Eine Webseite, die Amazon S3-Objekte mithilfe eines -SDK AWS auflistet
- Erste Schritte mit Amazon S3-Buckets und -Objekten unter Verwendung eines - AWS SDK
- Erste Schritte mit der Verschlüsselung für Amazon S3-Objekte mithilfe eines - AWS SDK
- Erste Schritte mit Tags für Amazon S3-Objekte unter Verwendung eines - AWS SDK
- Arbeiten mit Amazon S3-Objektsperrfunktionen unter Verwendung eines - AWS SDK
- Verwalten von Zugriffskontrolllisten (ACLs) für Amazon S3-Buckets mithilfe eines - AWS SDK
- Versionierte Amazon S3-Objekte in Batches mit einer Lambda-Funktion mithilfe eines AWS -SDK verwalten
- Analysieren von Amazon S3-URIs mit einem AWS -SDK
- Ausführen einer mehrteiligen Kopie eines Amazon S3-Objekts mithilfe eines - AWS SDK
- Durchführen eines mehrteiligen Uploads in ein Amazon S3-Objekt mithilfe eines - AWS SDK
- Hochladen oder Herunterladen großer Dateien in und von Amazon S3 mithilfe eines - AWS SDK
- Hochladen eines Streams unbekannter Größe in ein Amazon S3-Objekt mithilfe eines - AWS SDK
- Verwenden von Prüfsummen für die Arbeit mit einem Amazon S3-Objekt unter Verwendung eines - AWS SDK
- Arbeiten mit versionierten Amazon S3-Objekten unter Verwendung eines - AWS SDK

Erstellen einer vorsignierten URL für Amazon S3 mithilfe eines - AWS SDK

Die folgenden Codebeispiele veranschaulichen, wie Sie eine vorsignierte URL für Amazon S3 erstellen und ein Objekt hochladen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Generieren Sie eine vorsignierte URL, die für einen begrenzten Zeitraum eine Amazon-S3-Aktion ausführen kann.

```
    using System;
    using Amazon;
    using Amazon.S3;
    using Amazon.S3.Model;

    public class GenPresignedUrl
    {
        public static void Main()
        {
            const string bucketName = "doc-example-bucket";
            const string objectKey = "sample.txt";

            // Specify how long the presigned URL lasts, in hours
            const double timeoutDuration = 12;

            // Specify the AWS Region of your Amazon S3 bucket. If it is
            // different from the Region defined for the default user,
            // pass the Region to the constructor for the client. For
            // example: new AmazonS3Client(RegionEndpoint.USEast1);

            // If using the Region us-east-1, and server-side encryption with AWS
KMS, you must specify Signature Version 4.
            // Region us-east-1 defaults to Signature Version 2 unless explicitly
set to Version 4 as shown below.
```

```
            // For more details, see https://docs.aws.amazon.com/AmazonS3/latest/
userguide/UsingAWSSDK.html#specify-signature-version
            // and https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/Amazon/
TAWSConfigsS3.html
            AWSConfigsS3.UseSignatureVersion4 = true;
            IAmazonS3 s3Client = new AmazonS3Client(RegionEndpoint.USEast1);

            string urlString = GeneratePresignedURL(s3Client, bucketName,
 objectKey, timeoutDuration);
            Console.WriteLine($"The generated URL is: {urlString}.");
        }

        /// <summary>
        /// Generate a presigned URL that can be used to access the file named
        /// in the objectKey parameter for the amount of time specified in the
        /// duration parameter.
        /// </summary>
        /// <param name="client">An initialized S3 client object used to call
        /// the GetPresignedUrl method.</param>
        /// <param name="bucketName">The name of the S3 bucket containing the
        /// object for which to create the presigned URL.</param>
        /// <param name="objectKey">The name of the object to access with the
        /// presigned URL.</param>
        /// <param name="duration">The length of time for which the presigned
        /// URL will be valid.</param>
        /// <returns>A string representing the generated presigned URL.</returns>
        public static string GeneratePresignedURL(IAmazonS3 client, string
 bucketName, string objectKey, double duration)
        {
            string urlString = string.Empty;
            try
            {
                var request = new GetPreSignedUrlRequest()
                {
                    BucketName = bucketName,
                    Key = objectKey,
                    Expires = DateTime.UtcNow.AddHours(duration),
                };
                urlString = client.GetPreSignedURL(request);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error:'{ex.Message}'");
            }
```

```
            return urlString;
        }
    }
```

Generieren Sie eine vorsignierte URL und führen Sie ein Upload mit dieser URL durch.

```
    using System;
    using System.IO;
    using System.Net.Http;
    using System.Threading.Tasks;
    using Amazon;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to upload an object to an Amazon Simple Storage
    /// Service (Amazon S3) bucket using a presigned URL. The code first
    /// creates a presigned URL and then uses it to upload an object to an
    /// Amazon S3 bucket using that URL.
    /// </summary>
    public class UploadUsingPresignedURL
    {
        private static HttpClient httpClient = new HttpClient();

        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "samplefile.txt";
            string filePath = $"source\\{keyName}";

            // Specify how long the signed URL will be valid in hours.
            double timeoutDuration = 12;

            // Specify the AWS Region of your Amazon S3 bucket. If it is
            // different from the Region defined for the default user,
            // pass the Region to the constructor for the client. For
            // example: new AmazonS3Client(RegionEndpoint.USEast1);

            // If using the Region us-east-1, and server-side encryption with AWS
 KMS, you must specify Signature Version 4.
```

```
            // Region us-east-1 defaults to Signature Version 2 unless explicitly
 set to Version 4 as shown below.
            // For more details, see https://docs.aws.amazon.com/AmazonS3/latest/
userguide/UsingAWSSDK.html#specify-signature-version
            // and https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/Amazon/
TAWSConfigsS3.html
            AWSConfigsS3.UseSignatureVersion4 = true;
            IAmazonS3 client = new AmazonS3Client(RegionEndpoint.USEast1);

            var url = GeneratePreSignedURL(client, bucketName, keyName,
 timeoutDuration);
            var success = await UploadObject(filePath, url);

            if (success)
            {
                Console.WriteLine("Upload succeeded.");
            }
            else
            {
                Console.WriteLine("Upload failed.");
            }
        }

        /// <summary>
        /// Uploads an object to an Amazon S3 bucket using the presigned URL
 passed in
        /// the url parameter.
        /// </summary>
        /// <param name="filePath">The path (including file name) to the local
        /// file you want to upload.</param>
        /// <param name="url">The presigned URL that will be used to upload the
        /// file to the Amazon S3 bucket.</param>
        /// <returns>A Boolean value indicating the success or failure of the
        /// operation, based on the HttpWebResponse.</returns>
        public static async Task<bool> UploadObject(string filePath, string url)
        {
            using var streamContent = new StreamContent(
                new FileStream(filePath, FileMode.Open, FileAccess.Read));

            var response = await httpClient.PutAsync(url, streamContent);
            return response.IsSuccessStatusCode;
        }

        /// <summary>
```

```
        /// Generates a presigned URL which will be used to upload an object to
        /// an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// GetPreSignedURL.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket to which
 the
        /// presigned URL will point.</param>
        /// <param name="objectKey">The name of the file that will be uploaded.</
param>
        /// <param name="duration">How long (in hours) the presigned URL will
        /// be valid.</param>
        /// <returns>The generated URL.</returns>
        public static string GeneratePreSignedURL(
            IAmazonS3 client,
            string bucketName,
            string objectKey,
            double duration)
        {
            var request = new GetPreSignedUrlRequest
            {
                BucketName = bucketName,
                Key = objectKey,
                Verb = HttpVerb.PUT,
                Expires = DateTime.UtcNow.AddHours(duration),
            };

            string url = client.GetPreSignedURL(request);
            return url;
        }
    }
```

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die S3-Vorsignierungsaktionen umschließen.

```
// Presigner encapsulates the Amazon Simple Storage Service (Amazon S3) presign
 actions
// used in the examples.
// It contains PresignClient, a client that is used to presign requests to Amazon
 S3.
// Presigned requests contain temporary credentials and can be made from any HTTP
 client.
type Presigner struct {
 PresignClient *s3.PresignClient
}



// GetObject makes a presigned request that can be used to get an object from a
 bucket.
// The presigned request is valid for the specified number of seconds.
func (presigner Presigner) GetObject(
 bucketName string, objectKey string, lifetimeSecs int64)
 (*v4.PresignedHTTPRequest, error) {
 request, err := presigner.PresignClient.PresignGetObject(context.TODO(),
 &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 }, func(opts *s3.PresignOptions) {
  opts.Expires = time.Duration(lifetimeSecs * int64(time.Second))
 })
 if err != nil {
  log.Printf("Couldn't get a presigned request to get %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
```

```
 return request, err
}



// PutObject makes a presigned request that can be used to put an object in a
 bucket.
// The presigned request is valid for the specified number of seconds.
func (presigner Presigner) PutObject(
 bucketName string, objectKey string, lifetimeSecs int64)
 (*v4.PresignedHTTPRequest, error) {
 request, err := presigner.PresignClient.PresignPutObject(context.TODO(),
 &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 }, func(opts *s3.PresignOptions) {
  opts.Expires = time.Duration(lifetimeSecs * int64(time.Second))
 })
 if err != nil {
  log.Printf("Couldn't get a presigned request to put %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return request, err
}



// DeleteObject makes a presigned request that can be used to delete an object
 from a bucket.
func (presigner Presigner) DeleteObject(bucketName string, objectKey string)
 (*v4.PresignedHTTPRequest, error) {
 request, err := presigner.PresignClient.PresignDeleteObject(context.TODO(),
 &s3.DeleteObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't get a presigned request to delete object %v. Here's why:
 %v\n", objectKey, err)
 }
 return request, err
}
```

Führen Sie ein interaktives Beispiel aus, das vorsignierte URLs generiert und verwendet, um ein S3-Objekt hochzuladen, herunterzuladen und zu löschen.

```
// RunPresigningScenario is an interactive example that shows you how to get
 presigned
// HTTP requests that you can use to move data into and out of Amazon Simple
 Storage
// Service (Amazon S3). The presigned requests contain temporary credentials and
 can
// be used by an HTTP client.
//
// 1. Get a presigned request to put an object in a bucket.
// 2. Use the net/http package to use the presigned request to upload a local
 file to the bucket.
// 3. Get a presigned request to get an object from a bucket.
// 4. Use the net/http package to use the presigned request to download the
 object to a local file.
// 5. Get a presigned request to delete an object from a bucket.
// 6. Use the net/http package to use the presigned request to delete the object.
//
// This example creates an Amazon S3 presign client from the specified sdkConfig
 so that
// you can replace it with a mocked or stubbed config for unit testing.
//
// It uses a questioner from the `demotools` package to get input during the
 example.
// This package can be found in the ..\..\demotools folder of this repo.
//
// It uses an IHttpRequester interface to abstract HTTP requests so they can be
 mocked
// during testing.
func RunPresigningScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner, httpRequester IHttpRequester) {
 defer func() {
  if r := recover(); r != nil {
   fmt.Printf("Something went wrong with the demo.")
  }
 }()

 log.Println(strings.Repeat("-", 88))
```

```
log.Println("Welcome to the Amazon S3 presigning demo.")
log.Println(strings.Repeat("-", 88))

s3Client := s3.NewFromConfig(sdkConfig)
bucketBasics := actions.BucketBasics{S3Client: s3Client}
presignClient := s3.NewPresignClient(s3Client)
presigner := actions.Presigner{PresignClient: presignClient}

bucketName := questioner.Ask("We'll need a bucket. Enter a name for a bucket "+
 "you own or one you want to create:", demotools.NotEmpty{})
bucketExists, err := bucketBasics.BucketExists(bucketName)
if err != nil {
 panic(err)
}
if !bucketExists {
 err = bucketBasics.CreateBucket(bucketName, sdkConfig.Region)
 if err != nil {
  panic(err)
 } else {
  log.Println("Bucket created.")
 }
}
log.Println(strings.Repeat("-", 88))

log.Printf("Let's presign a request to upload a file to your bucket.")
uploadFilename := questioner.Ask("Enter the path to a file you want to upload:",
 demotools.NotEmpty{})
uploadKey := questioner.Ask("What would you like to name the uploaded object?",
 demotools.NotEmpty{})
uploadFile, err := os.Open(uploadFilename)
if err != nil {
 panic(err)
}
defer uploadFile.Close()
presignedPutRequest, err := presigner.PutObject(bucketName, uploadKey, 60)
if err != nil {
 panic(err)
}
log.Printf("Got a presigned %v request to URL:\n\t%v\n",
presignedPutRequest.Method,
 presignedPutRequest.URL)
log.Println("Using net/http to send the request...")
info, err := uploadFile.Stat()
if err != nil {
```

```
 panic(err)
}
putResponse, err := httpRequester.Put(presignedPutRequest.URL, info.Size(),
uploadFile)
if err != nil {
 panic(err)
}
log.Printf("%v object %v with presigned URL returned %v.",
presignedPutRequest.Method,
 uploadKey, putResponse.StatusCode)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's presign a request to download the object.")
questioner.Ask("Press Enter when you're ready.")
presignedGetRequest, err := presigner.GetObject(bucketName, uploadKey, 60)
if err != nil {
 panic(err)
}
log.Printf("Got a presigned %v request to URL:\n\t%v\n",
presignedGetRequest.Method,
 presignedGetRequest.URL)
log.Println("Using net/http to send the request...")
getResponse, err := httpRequester.Get(presignedGetRequest.URL)
if err != nil {
 panic(err)
}
log.Printf("%v object %v with presigned URL returned %v.",
presignedGetRequest.Method,
 uploadKey, getResponse.StatusCode)
defer getResponse.Body.Close()
downloadBody, err := io.ReadAll(getResponse.Body)
if err != nil {
 panic(err)
}
log.Printf("Downloaded %v bytes. Here are the first 100 of them:\n",
len(downloadBody))
log.Println(strings.Repeat("-", 88))
log.Println(string(downloadBody[:100]))
log.Println(strings.Repeat("-", 88))

log.Println("Let's presign a request to delete the object.")
questioner.Ask("Press Enter when you're ready.")
presignedDelRequest, err := presigner.DeleteObject(bucketName, uploadKey)
if err != nil {
```

```
  panic(err)
 }
 log.Printf("Got a presigned %v request to URL:\n\t%v\n",
 presignedDelRequest.Method,
  presignedDelRequest.URL)
 log.Println("Using net/http to send the request...")
 delResponse, err := httpRequester.Delete(presignedDelRequest.URL)
 if err != nil {
  panic(err)
 }
 log.Printf("%v object %v with presigned URL returned %v.\n",
 presignedDelRequest.Method,
  uploadKey, delResponse.StatusCode)
 log.Println(strings.Repeat("-", 88))

 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Definieren Sie einen HTTP-Request-Wrapper, der im Beispiel verwendet wird, um HTTP-Anfragen zu stellen.

```
// IHttpRequester abstracts HTTP requests into an interface so it can be mocked
 during
// unit testing.
type IHttpRequester interface {
 Get(url string) (resp *http.Response, err error)
 Put(url string, contentLength int64, body io.Reader) (resp *http.Response, err
 error)
 Delete(url string) (resp *http.Response, err error)
}

// HttpRequester uses the net/http package to make HTTP requests during the
 scenario.
type HttpRequester struct{}

func (httpReq HttpRequester) Get(url string) (resp *http.Response, err error) {
 return http.Get(url)
}
```

```
func (httpReq HttpRequester) Put(url string, contentLength int64, body io.Reader)
 (resp *http.Response, err error) {
 putRequest, err := http.NewRequest("PUT", url, body)
 if err != nil {
  return nil, err
 }
 putRequest.ContentLength = contentLength
 return http.DefaultClient.Do(putRequest)
}
func (httpReq HttpRequester) Delete(url string) (resp *http.Response, err error)
 {
 delRequest, err := http.NewRequest("DELETE", url, nil)
 if err != nil {
  return nil, err
 }
 return http.DefaultClient.Do(delRequest)
}
```

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Generieren Sie eine vorsignierte URL für ein Objekt und laden Sie sie dann herunter (GET-Anforderung).

Importiert.

```
import com.example.s3.util.PresignUrlUtils;
import org.slf4j.Logger;
import software.amazon.awssdk.http.HttpExecuteRequest;
import software.amazon.awssdk.http.HttpExecuteResponse;
import software.amazon.awssdk.http.SdkHttpClient;
import software.amazon.awssdk.http.SdkHttpMethod;
```

```
import software.amazon.awssdk.http.SdkHttpRequest;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import
 software.amazon.awssdk.services.s3.presigner.model.GetObjectPresignRequest;
import
 software.amazon.awssdk.services.s3.presigner.model.PresignedGetObjectRequest;
import software.amazon.awssdk.utils.IoUtils;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URISyntaxException;
import java.net.URL;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.nio.file.Paths;
import java.time.Duration;
import java.util.UUID;
```

Generieren Sie die URL.

```
   /* Create a pre-signed URL to download an object in a subsequent GET request.
 */
   public String createPresignedGetUrl(String bucketName, String keyName) {
       try (S3Presigner presigner = S3Presigner.create()) {

           GetObjectRequest objectRequest = GetObjectRequest.builder()
                   .bucket(bucketName)
                   .key(keyName)
                   .build();

           GetObjectPresignRequest presignRequest =
 GetObjectPresignRequest.builder()
                   .signatureDuration(Duration.ofMinutes(10))  // The URL will
 expire in 10 minutes.
```

```
                    .getObjectRequest(objectRequest)
                    .build();

            PresignedGetObjectRequest presignedRequest =
presigner.presignGetObject(presignRequest);
            logger.info("Presigned URL: [{}]",
presignedRequest.url().toString());
            logger.info("HTTP method: [{}]",
presignedRequest.httpRequest().method());

            return presignedRequest.url().toExternalForm();
        }
    }
```

Laden Sie das Objekt mit einem der folgenden drei Ansätze herunter.

Verwenden Sie die Klasse JDK `HttpURLConnection` (seit v1.1), um den Download durchzuführen.

```
    /* Use the JDK HttpURLConnection (since v1.1) class to do the download. */
    public byte[] useHttpUrlConnectionToGet(String presignedUrlString) {
        ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream(); // Capture the response body to a byte array.

        try {
            URL presignedUrl = new URL(presignedUrlString);
            HttpURLConnection connection = (HttpURLConnection)
presignedUrl.openConnection();
            connection.setRequestMethod("GET");
            // Download the result of executing the request.
            try (InputStream content = connection.getInputStream()) {
                IoUtils.copy(content, byteArrayOutputStream);
            }
            logger.info("HTTP response code is " + connection.getResponseCode());

        } catch (S3Exception | IOException e) {
            logger.error(e.getMessage(), e);
        }
        return byteArrayOutputStream.toByteArray();
    }
```

Verwenden Sie die Klasse JDK `HttpClient` (seit v11), um den Download durchzuführen.

```
   /* Use the JDK HttpClient (since v11) class to do the download. */
   public byte[] useHttpClientToGet(String presignedUrlString) {
       ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream(); // Capture the response body to a byte array.

       HttpRequest.Builder requestBuilder = HttpRequest.newBuilder();
       HttpClient httpClient = HttpClient.newHttpClient();
       try {
           URL presignedUrl = new URL(presignedUrlString);
           HttpResponse<InputStream> response = httpClient.send(requestBuilder
                           .uri(presignedUrl.toURI())
                           .GET()
                           .build(),
                   HttpResponse.BodyHandlers.ofInputStream());

           IoUtils.copy(response.body(), byteArrayOutputStream);

           logger.info("HTTP response code is " + response.statusCode());

       } catch (URISyntaxException | InterruptedException | IOException e) {
           logger.error(e.getMessage(), e);
       }
       return byteArrayOutputStream.toByteArray();
   }
```

Verwenden Sie die `SdkHttpClient` Klasse AWS SDK for Java, um den Download durchzuführen.

```
   /* Use the AWS SDK for Java SdkHttpClient class to do the download. */
   public byte[] useSdkHttpClientToPut(String presignedUrlString) {

       ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream(); // Capture the response body to a byte array.
       try {
           URL presignedUrl = new URL(presignedUrlString);
           SdkHttpRequest request = SdkHttpRequest.builder()
                   .method(SdkHttpMethod.GET)
                   .uri(presignedUrl.toURI())
                   .build();
```

```
            HttpExecuteRequest executeRequest = HttpExecuteRequest.builder()
                    .request(request)
                    .build();

            try (SdkHttpClient sdkHttpClient = ApacheHttpClient.create()) {
                HttpExecuteResponse response =
sdkHttpClient.prepareRequest(executeRequest).call();
                response.responseBody().ifPresentOrElse(
                        abortableInputStream -> {
                            try {
                                IoUtils.copy(abortableInputStream,
byteArrayOutputStream);
                            } catch (IOException e) {
                                throw new RuntimeException(e);
                            }
                        },
                        () -> logger.error("No response body."));

                logger.info("HTTP Response code is {}",
response.httpResponse().statusCode());
            }
        } catch (URISyntaxException | IOException e) {
            logger.error(e.getMessage(), e);
        }
        return byteArrayOutputStream.toByteArray();
    }
```

Generieren Sie eine vorsignierte URL für einen Upload und laden Sie dann eine Datei hoch (PUT-Anforderung).

Importiert.

```
import com.example.s3.util.PresignUrlUtils;
import org.slf4j.Logger;
import software.amazon.awssdk.core.internal.sync.FileContentStreamProvider;
import software.amazon.awssdk.http.HttpExecuteRequest;
import software.amazon.awssdk.http.HttpExecuteResponse;
import software.amazon.awssdk.http.SdkHttpClient;
import software.amazon.awssdk.http.SdkHttpMethod;
import software.amazon.awssdk.http.SdkHttpRequest;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.services.s3.S3Client;
```

```
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import
 software.amazon.awssdk.services.s3.presigner.model.PresignedPutObjectRequest;
import
 software.amazon.awssdk.services.s3.presigner.model.PutObjectPresignRequest;

import java.io.File;
import java.io.IOException;
import java.io.OutputStream;
import java.io.RandomAccessFile;
import java.net.HttpURLConnection;
import java.net.URISyntaxException;
import java.net.URL;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.nio.ByteBuffer;
import java.nio.channels.FileChannel;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Duration;
import java.util.Map;
import java.util.UUID;
```

Generieren Sie die URL.

```
   /* Create a presigned URL to use in a subsequent PUT request */
   public String createPresignedUrl(String bucketName, String keyName,
Map<String, String> metadata) {
       try (S3Presigner presigner = S3Presigner.create()) {

           PutObjectRequest objectRequest = PutObjectRequest.builder()
                   .bucket(bucketName)
                   .key(keyName)
                   .metadata(metadata)
                   .build();

           PutObjectPresignRequest presignRequest =
PutObjectPresignRequest.builder()
```

```
                    .signatureDuration(Duration.ofMinutes(10))  // The URL
 expires in 10 minutes.
                    .putObjectRequest(objectRequest)
                    .build();


            PresignedPutObjectRequest presignedRequest =
 presigner.presignPutObject(presignRequest);
            String myURL = presignedRequest.url().toString();
            logger.info("Presigned URL to upload a file to: [{}]", myURL);
            logger.info("HTTP method: [{}]",
 presignedRequest.httpRequest().method());

            return presignedRequest.url().toExternalForm();
        }
    }
```

Laden Sie ein Dateiobjekt hoch, indem Sie einen der folgenden drei Ansätze verwenden.

Verwenden Sie die Klasse JDK `HttpURLConnection` (seit v1.1), um den Upload durchzuführen.

```
    /* Use the JDK HttpURLConnection (since v1.1) class to do the upload. */
    public void useHttpUrlConnectionToPut(String presignedUrlString, File
 fileToPut, Map<String, String> metadata) {
        logger.info("Begin [{}] upload", fileToPut.toString());
        try {
            URL presignedUrl = new URL(presignedUrlString);
            HttpURLConnection connection = (HttpURLConnection)
 presignedUrl.openConnection();
            connection.setDoOutput(true);
            metadata.forEach((k, v) -> connection.setRequestProperty("x-amz-
meta-" + k, v));
            connection.setRequestMethod("PUT");
            OutputStream out = connection.getOutputStream();

            try (RandomAccessFile file = new RandomAccessFile(fileToPut, "r");
                 FileChannel inChannel = file.getChannel()) {
                ByteBuffer buffer = ByteBuffer.allocate(8192); //Buffer size is
 8k

                while (inChannel.read(buffer) > 0) {
```

```
                    buffer.flip();
                    for (int i = 0; i < buffer.limit(); i++) {
                        out.write(buffer.get());
                    }
                    buffer.clear();
                }
            } catch (IOException e) {
                logger.error(e.getMessage(), e);
            }

            out.close();
            connection.getResponseCode();
            logger.info("HTTP response code is " + connection.getResponseCode());

        } catch (S3Exception | IOException e) {
            logger.error(e.getMessage(), e);
        }
    }
```

Verwenden Sie die Klasse JDK `HttpClient` (seit v11), um den Upload durchzuführen.

```
    /* Use the JDK HttpClient (since v11) class to do the upload. */
    public void useHttpClientToPut(String presignedUrlString, File fileToPut,
Map<String, String> metadata) {
        logger.info("Begin [{}] upload", fileToPut.toString());

        HttpRequest.Builder requestBuilder = HttpRequest.newBuilder();
        metadata.forEach((k, v) -> requestBuilder.header("x-amz-meta-" + k, v));

        HttpClient httpClient = HttpClient.newHttpClient();
        try {
            final HttpResponse<Void> response = httpClient.send(requestBuilder
                            .uri(new URL(presignedUrlString).toURI())

.PUT(HttpRequest.BodyPublishers.ofFile(Path.of(fileToPut.toURI())))
                            .build(),
                    HttpResponse.BodyHandlers.discarding());

            logger.info("HTTP response code is " + response.statusCode());

        } catch (URISyntaxException | InterruptedException | IOException e) {
            logger.error(e.getMessage(), e);
```

```
        }
    }
```

Verwenden Sie die `SdkHttpClient` Klasse AWS für Java V2, um den Upload durchzuführen.

```
   /* Use the AWS SDK for Java V2 SdkHttpClient class to do the upload. */
   public void useSdkHttpClientToPut(String presignedUrlString, File fileToPut,
Map<String, String> metadata) {
       logger.info("Begin [{}] upload", fileToPut.toString());

       try {
           URL presignedUrl = new URL(presignedUrlString);

           SdkHttpRequest.Builder requestBuilder = SdkHttpRequest.builder()
                   .method(SdkHttpMethod.PUT)
                   .uri(presignedUrl.toURI());
           // Add headers
           metadata.forEach((k, v) -> requestBuilder.putHeader("x-amz-meta-" +
k, v));
           // Finish building the request.
           SdkHttpRequest request = requestBuilder.build();

           HttpExecuteRequest executeRequest = HttpExecuteRequest.builder()
                   .request(request)
                   .contentStreamProvider(new
FileContentStreamProvider(fileToPut.toPath()))
                   .build();

           try (SdkHttpClient sdkHttpClient = ApacheHttpClient.create()) {
               HttpExecuteResponse response =
sdkHttpClient.prepareRequest(executeRequest).call();
               logger.info("Response code: {}",
response.httpResponse().statusCode());
           }
       } catch (URISyntaxException | IOException e) {
           logger.error(e.getMessage(), e);
       }
   }
```

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie eine vorsignierte URL, um ein Objekt in einen Bucket hochzuladen.

```
import https from "https";
import { PutObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { fromIni } from "@aws-sdk/credential-providers";
import { HttpRequest } from "@smithy/protocol-http";
import {
  getSignedUrl,
  S3RequestPresigner,
} from "@aws-sdk/s3-request-presigner";
import { parseUrl } from "@smithy/url-parser";
import { formatUrl } from "@aws-sdk/util-format-url";
import { Hash } from "@smithy/hash-node";

const createPresignedUrlWithoutClient = async ({ region, bucket, key }) => {
  const url = parseUrl(`https://${bucket}.s3.${region}.amazonaws.com/${key}`);
  const presigner = new S3RequestPresigner({
    credentials: fromIni(),
    region,
    sha256: Hash.bind(null, "sha256"),
  });

  const signedUrlObject = await presigner.presign(
    new HttpRequest({ ...url, method: "PUT" }),
  );
  return formatUrl(signedUrlObject);
};

const createPresignedUrlWithClient = ({ region, bucket, key }) => {
  const client = new S3Client({ region });
  const command = new PutObjectCommand({ Bucket: bucket, Key: key });
  return getSignedUrl(client, command, { expiresIn: 3600 });
};
```

```
function put(url, data) {
  return new Promise((resolve, reject) => {
    const req = https.request(
      url,
      { method: "PUT", headers: { "Content-Length": new Blob([data]).size } },
      (res) => {
        let responseBody = "";
        res.on("data", (chunk) => {
          responseBody += chunk;
        });
        res.on("end", () => {
          resolve(responseBody);
        });
      },
    );
    req.on("error", (err) => {
      reject(err);
    });
    req.write(data);
    req.end();
  });
}

export const main = async () => {
  const REGION = "us-east-1";
  const BUCKET = "example_bucket";
  const KEY = "example_file.txt";

  // There are two ways to generate a presigned URL.
  // 1. Use createPresignedUrl without the S3 client.
  // 2. Use getSignedUrl in conjunction with the S3 client and GetObjectCommand.
  try {
    const noClientUrl = await createPresignedUrlWithoutClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });

    const clientUrl = await createPresignedUrlWithClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });
```

```
    // After you get the presigned URL, you can provide your own file
    // data. Refer to put() above.
    console.log("Calling PUT using presigned URL without client");
    await put(noClientUrl, "Hello World");

    console.log("Calling PUT using presigned URL with client");
    await put(clientUrl, "Hello World");

    console.log("\nDone. Check your S3 console.");
  } catch (err) {
    console.error(err);
  }
};
```

Erstellen Sie eine vorsignierte URL, um ein Objekt aus einem Bucket herunterzuladen.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { fromIni } from "@aws-sdk/credential-providers";
import { HttpRequest } from "@smithy/protocol-http";
import {
  getSignedUrl,
  S3RequestPresigner,
} from "@aws-sdk/s3-request-presigner";
import { parseUrl } from "@smithy/url-parser";
import { formatUrl } from "@aws-sdk/util-format-url";
import { Hash } from "@smithy/hash-node";

const createPresignedUrlWithoutClient = async ({ region, bucket, key }) => {
  const url = parseUrl(`https://${bucket}.s3.${region}.amazonaws.com/${key}`);
  const presigner = new S3RequestPresigner({
    credentials: fromIni(),
    region,
    sha256: Hash.bind(null, "sha256"),
  });

  const signedUrlObject = await presigner.presign(new HttpRequest(url));
  return formatUrl(signedUrlObject);
};

const createPresignedUrlWithClient = ({ region, bucket, key }) => {
  const client = new S3Client({ region });
```

```
  const command = new GetObjectCommand({ Bucket: bucket, Key: key });
  return getSignedUrl(client, command, { expiresIn: 3600 });
};

export const main = async () => {
  const REGION = "us-east-1";
  const BUCKET = "example_bucket";
  const KEY = "example_file.jpg";

  try {
    const noClientUrl = await createPresignedUrlWithoutClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });

    const clientUrl = await createPresignedUrlWithClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });

    console.log("Presigned URL without client");
    console.log(noClientUrl);
    console.log("\n");

    console.log("Presigned URL with client");
    console.log(clientUrl);
  } catch (err) {
    console.error(err);
  }
};
```

- Weitere Informationen finden Sie im AWS SDK for JavaScript -Entwicklerhandbuch.

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie eine vorsignierte `GetObject`-Anfrage und verwenden Sie die URL, um ein Objekt herunterzuladen.

```
suspend fun getObjectPresigned(s3: S3Client, bucketName: String, keyName:
 String): String {
    // Create a GetObjectRequest.
    val unsignedRequest = GetObjectRequest {
        bucket = bucketName
        key = keyName
    }

    // Presign the GetObject request.
    val presignedRequest = s3.presignGetObject(unsignedRequest, 24.hours)

    // Use the URL from the presigned HttpRequest in a subsequent HTTP GET
 request to retrieve the object.
    val objectContents = URL(presignedRequest.url.toString()).readText()

    return objectContents
}
```

Erstellen Sie eine `GetObject` vorsignierte Anforderung mit erweiterten Optionen.

```
suspend fun getObjectPresignedMoreOptions(s3: S3Client, bucketName: String,
 keyName: String): HttpRequest {
    // Create a GetObjectRequest.
    val unsignedRequest = GetObjectRequest {
        bucket = bucketName
        key = keyName
    }
```

```
    // Presign the GetObject request.
    val presignedRequest = s3.presignGetObject(unsignedRequest, signer =
 CrtAwsSigner) {
        signingDate = Instant.now() + 12.hours // Presigned request can be used
 12 hours from now.
        algorithm = AwsSigningAlgorithm.SIGV4_ASYMMETRIC
        signatureType = AwsSignatureType.HTTP_REQUEST_VIA_QUERY_PARAMS
        expiresAfter = 8.hours // Presigned request expires 8 hours later.
    }
    return presignedRequest
}
```

Erstellen Sie eine vorsignierte `PutObject`-Anfrage und verwenden Sie sie, um ein Objekt hochzuladen.

```
suspend fun putObjectPresigned(s3: S3Client, bucketName: String, keyName: String,
 content: String) {
    // Create a PutObjectRequest.
    val unsignedRequest = PutObjectRequest {
        bucket = bucketName
        key = keyName
    }

    // Presign the request.
    val presignedRequest = s3.presignPutObject(unsignedRequest, 24.hours)

    // Use the URL and any headers from the presigned HttpRequest in a subsequent
 HTTP PUT request to retrieve the object.
    // Create a PUT request using the OKHttpClient API.
    val putRequest = Request
        .Builder()
        .url(presignedRequest.url.toString())
        .apply {
            presignedRequest.headers.forEach { key, values ->
                header(key, values.joinToString(", "))
            }
        }
        .put(content.toRequestBody())
        .build()

    val response = OkHttpClient().newCall(putRequest).execute()
    assert(response.isSuccessful)
```

```
}
```

- Weitere Informationen finden Sie im Entwicklerhandbuch zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Generieren Sie eine vorsignierte URL, die für einen begrenzten Zeitraum eine S3-Aktion ausführen kann. Verwenden Sie das Anforderungspaket, um eine Anforderung mit der URL zu stellen.

```
import argparse
import logging
import boto3
from botocore.exceptions import ClientError
import requests

logger = logging.getLogger(__name__)


def generate_presigned_url(s3_client, client_method, method_parameters,
 expires_in):
    """
    Generate a presigned Amazon S3 URL that can be used to perform an action.

    :param s3_client: A Boto3 Amazon S3 client.
    :param client_method: The name of the client method that the URL performs.
    :param method_parameters: The parameters of the specified client method.
    :param expires_in: The number of seconds the presigned URL is valid for.
    :return: The presigned URL.
    """
    try:
        url = s3_client.generate_presigned_url(
```

```
            ClientMethod=client_method, Params=method_parameters,
 ExpiresIn=expires_in
        )
        logger.info("Got presigned URL: %s", url)
    except ClientError:
        logger.exception(
            "Couldn't get a presigned URL for client method '%s'.", client_method
        )
        raise
    return url


def usage_demo():
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Welcome to the Amazon S3 presigned URL demo.")
    print("-" * 88)

    parser = argparse.ArgumentParser()
    parser.add_argument("bucket", help="The name of the bucket.")
    parser.add_argument(
        "key",
        help="For a GET operation, the key of the object in Amazon S3. For a "
        "PUT operation, the name of a file to upload.",
    )
    parser.add_argument("action", choices=("get", "put"), help="The action to
 perform.")
    args = parser.parse_args()

    s3_client = boto3.client("s3")
    client_action = "get_object" if args.action == "get" else "put_object"
    url = generate_presigned_url(
        s3_client, client_action, {"Bucket": args.bucket, "Key": args.key}, 1000
    )

    print("Using the Requests package to send a request to the URL.")
    response = None
    if args.action == "get":
        response = requests.get(url)
    elif args.action == "put":
        print("Putting data to the URL.")
        try:
            with open(args.key, "r") as object_file:
```

```
                object_text = object_file.read()
            response = requests.put(url, data=object_text)
        except FileNotFoundError:
            print(
                f"Couldn't find {args.key}. For a PUT operation, the key must be
 the "
                f"name of a file that exists on your computer."
            )

    if response is not None:
        print("Got response:")
        print(f"Status: {response.status_code}")
        print(response.text)

    print("-" * 88)


if __name__ == "__main__":
    usage_demo()
```

Generieren Sie eine vorsignierte POST-Anforderung zum Hochladen einer Datei.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def generate_presigned_post(self, object_key, expires_in):
        """
        Generate a presigned Amazon S3 POST request to upload a file.
        A presigned POST can be used for a limited time to let someone without an
 AWS
        account upload a file to a bucket.
```

```
        :param object_key: The object key to identify the uploaded object.
        :param expires_in: The number of seconds the presigned POST is valid.
        :return: A dictionary that contains the URL and form fields that contain
                 required access data.
        """
        try:
            response = self.bucket.meta.client.generate_presigned_post(
                Bucket=self.bucket.name, Key=object_key, ExpiresIn=expires_in
            )
            logger.info("Got presigned POST URL: %s", response["url"])
        except ClientError:
            logger.exception(
                "Couldn't get a presigned POST URL for bucket '%s' and object
'%s'",
                self.bucket.name,
                object_key,
            )
            raise
        return response
```

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"
require "net/http"

# Creates a presigned URL that can be used to upload content to an object.
#
# @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
# @param object_key [String] The key to give the uploaded object.
# @return [URI, nil] The parsed URI if successful; otherwise nil.
```

```
def get_presigned_url(bucket, object_key)
  url = bucket.object(object_key).presigned_url(:put)
  puts "Created presigned URL: #{url}"
  URI(url)
rescue Aws::Errors::ServiceError => e
  puts "Couldn't create presigned URL for #{bucket.name}:#{object_key}. Here's
 why: #{e.message}"
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-file.txt"
  object_content = "This is the content of my-file.txt."

  bucket = Aws::S3::Bucket.new(bucket_name)
  presigned_url = get_presigned_url(bucket, object_key)
  return unless presigned_url

  response = Net::HTTP.start(presigned_url.host) do |http|
    http.send_request("PUT", presigned_url.request_uri, object_content,
 "content_type" => "")
  end

  case response
  when Net::HTTPSuccess
    puts "Content uploaded!"
  else
    puts response.value
  end
end

run_demo if $PROGRAM_NAME == __FILE__
```

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Vorsignieranforderungen für GET- und PUT-S3-Objekte.

```
async fn get_object(
    client: &Client,
    bucket: &str,
    object: &str,
    expires_in: u64,
) -> Result<(), Box<dyn Error>> {
    let expires_in = Duration::from_secs(expires_in);
    let presigned_request = client
        .get_object()
        .bucket(bucket)
        .key(object)
        .presigned(PresigningConfig::expires_in(expires_in)?)
        .await?;

    println!("Object URI: {}", presigned_request.uri());

    Ok(())
}

async fn put_object(
    client: &Client,
    bucket: &str,
    object: &str,
    expires_in: u64,
) -> Result<(), Box<dyn Error>> {
    let expires_in = Duration::from_secs(expires_in);

    let presigned_request = client
        .put_object()
        .bucket(bucket)
        .key(object)
```

```
        .presigned(PresigningConfig::expires_in(expires_in)?)
        .await?;

    println!("Object URI: {}", presigned_request.uri());

    Ok(())
}
```

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Eine Webseite, die Amazon S3-Objekte mithilfe eines -SDK AWS auflistet

Im folgenden Codebeispiel wird veranschaulicht, wie Sie Amazon-S3-Objekte auf einer Webseite auflisten.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Der folgende Code ist die relevante React-Komponente, die Aufrufe an das AWS SDK tätigt. Eine ausführbare Version der Anwendung, die diese Komponente enthält, finden Sie unter dem vorherigen GitHub Link.

```
import { useEffect, useState } from "react";
import {
  ListObjectsCommand,
  ListObjectsCommandOutput,
  S3Client,
} from "@aws-sdk/client-s3";
import { fromCognitoIdentityPool } from "@aws-sdk/credential-providers";
import "./App.css";
```

```
function App() {
  const [objects, setObjects] = useState<
    Required<ListObjectsCommandOutput>["Contents"]
  >([]);

  useEffect(() => {
    const client = new S3Client({
      region: "us-east-1",
      // Unless you have a public bucket, you'll need access to a private bucket.
      // One way to do this is to create an Amazon Cognito identity pool, attach
 a role to the pool,
      // and grant the role access to the 's3:GetObject' action.
      //
      // You'll also need to configure the CORS settings on the bucket to allow
 traffic from
      // this example site. Here's an example configuration that allows all
 origins. Don't
      // do this in production.
      //[
      //  {
      //    "AllowedHeaders": ["*"],
      //    "AllowedMethods": ["GET"],
      //    "AllowedOrigins": ["*"],
      //    "ExposeHeaders": [],
      //  },
      //]
      //
      credentials: fromCognitoIdentityPool({
        clientConfig: { region: "us-east-1" },
        identityPoolId: "<YOUR_IDENTITY_POOL_ID>",
      }),
    });
    const command = new ListObjectsCommand({ Bucket: "bucket-name" });
    client.send(command).then(({ Contents }) => setObjects(Contents || []));
  }, []);

  return (
    <div className="App">
      {objects.map((o) => (
        <div key={o.ETag}>{o.Key}</div>
      ))}
    </div>
  );
```

```
}

export default App;
```

- Weitere API-Informationen finden Sie unter ListObjects in der APIAWS SDK for JavaScript - Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte mit Amazon S3-Buckets und -Objekten unter Verwendung eines - AWS SDK

Die folgenden Code-Beispiele veranschaulichen Folgendes:

- Erstellen Sie einen Bucket und laden Sie eine Datei in ihn hoch.
- Laden Sie ein Objekt aus einem Bucket herunter.
- Kopieren Sie ein Objekt in einen Unterordner eines Buckets.
- Listen Sie die Objekte in einem Bucket auf.
- Löschen Sie die Bucket-Objekte und den Bucket.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
public class S3_Basics
{
    public static async Task Main()
    {
        // Create an Amazon S3 client object. The constructor uses the
```

```
            // default user installed on the system. To work with Amazon S3
            // features in a different AWS Region, pass the AWS Region as a
            // parameter to the client constructor.
            IAmazonS3 client = new AmazonS3Client();
            string bucketName = string.Empty;
            string filePath = string.Empty;
            string keyName = string.Empty;

            var sepBar = new string('-', Console.WindowWidth);

            Console.WriteLine(sepBar);
            Console.WriteLine("Amazon Simple Storage Service (Amazon S3) basic");
            Console.WriteLine("procedures. This application will:");
            Console.WriteLine("\n\t1. Create a bucket");
            Console.WriteLine("\n\t2. Upload an object to the new bucket");
            Console.WriteLine("\n\t3. Copy the uploaded object to a folder in the
 bucket");
            Console.WriteLine("\n\t4. List the items in the new bucket");
            Console.WriteLine("\n\t5. Delete all the items in the bucket");
            Console.WriteLine("\n\t6. Delete the bucket");
            Console.WriteLine(sepBar);

            // Create a bucket.
            Console.WriteLine($"\n{sepBar}");
            Console.WriteLine("\nCreate a new Amazon S3 bucket.\n");
            Console.WriteLine(sepBar);

            Console.Write("Please enter a name for the new bucket: ");
            bucketName = Console.ReadLine();

            var success = await S3Bucket.CreateBucketAsync(client, bucketName);
            if (success)
            {
                Console.WriteLine($"Successfully created bucket: {bucketName}.
\n");
            }
            else
            {
                Console.WriteLine($"Could not create bucket: {bucketName}.\n");
            }

            Console.WriteLine(sepBar);
            Console.WriteLine("Upload a file to the new bucket.");
            Console.WriteLine(sepBar);
```

```
            // Get the local path and filename for the file to upload.
            while (string.IsNullOrEmpty(filePath))
            {
                Console.Write("Please enter the path and filename of the file to
upload: ");
                filePath = Console.ReadLine();

                // Confirm that the file exists on the local computer.
                if (!File.Exists(filePath))
                {
                    Console.WriteLine($"Couldn't find {filePath}. Try again.\n");
                    filePath = string.Empty;
                }
            }

            // Get the file name from the full path.
            keyName = Path.GetFileName(filePath);

            success = await S3Bucket.UploadFileAsync(client, bucketName, keyName,
filePath);

            if (success)
            {
                Console.WriteLine($"Successfully uploaded {keyName} from
{filePath} to {bucketName}.\n");
            }
            else
            {
                Console.WriteLine($"Could not upload {keyName}.\n");
            }

            // Set the file path to an empty string to avoid overwriting the
            // file we just uploaded to the bucket.
            filePath = string.Empty;

            // Now get a new location where we can save the file.
            while (string.IsNullOrEmpty(filePath))
            {
                // First get the path to which the file will be downloaded.
                Console.Write("Please enter the path where the file will be
downloaded: ");
                filePath = Console.ReadLine();
```

```
                // Confirm that the file exists on the local computer.
                if (File.Exists($"{filePath}\\{keyName}"))
                {
                    Console.WriteLine($"Sorry, the file already exists in that
location.\n");
                    filePath = string.Empty;
                }
            }

            // Download an object from a bucket.
            success = await S3Bucket.DownloadObjectFromBucketAsync(client,
bucketName, keyName, filePath);

            if (success)
            {
                Console.WriteLine($"Successfully downloaded {keyName}.\n");
            }
            else
            {
                Console.WriteLine($"Sorry, could not download {keyName}.\n");
            }

            // Copy the object to a different folder in the bucket.
            string folderName = string.Empty;

            while (string.IsNullOrEmpty(folderName))
            {
                Console.Write("Please enter the name of the folder to copy your
object to: ");
                folderName = Console.ReadLine();
            }

            while (string.IsNullOrEmpty(keyName))
            {
                // Get the name to give to the object once uploaded.
                Console.Write("Enter the name of the object to copy: ");
                keyName = Console.ReadLine();
            }

            await S3Bucket.CopyObjectInBucketAsync(client, bucketName, keyName,
folderName);

            // List the objects in the bucket.
            await S3Bucket.ListBucketContentsAsync(client, bucketName);
```

```
            // Delete the contents of the bucket.
            await S3Bucket.DeleteBucketContentsAsync(client, bucketName);

            // Deleting the bucket too quickly after deleting its contents will
            // cause an error that the bucket isn't empty. So...
            Console.WriteLine("Press <Enter> when you are ready to delete the
bucket.");
            _ = Console.ReadLine();

            // Delete the bucket.
            await S3Bucket.DeleteBucketAsync(client, bucketName);
        }
    }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
###############################################################################
# function s3_getting_started
```

```
#
# This function creates, copies, and deletes S3 buckets and objects.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function s3_getting_started() {
  {
    if [ "$BUCKET_OPERATIONS_SOURCED" != "True" ]; then
      cd bucket-lifecycle-operations || exit

      source ./bucket_operations.sh
      cd ..
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the Amazon S3 getting started demo."
  echo_repeat "*" 88

  local bucket_name
  bucket_name=$(generate_random_name "doc-example-bucket")

  local region_code
  region_code=$(aws configure get region)

  if create_bucket -b "$bucket_name" -r "$region_code"; then
    echo "Created demo bucket named $bucket_name"
  else
    errecho "The bucket failed to create. This demo will exit."
    return 1
  fi

  local file_name
  while [ -z "$file_name" ]; do
    echo -n "Enter a file you want to upload to your bucket: "
    get_input
    file_name=$get_input_result

    if [ ! -f "$file_name" ]; then
      echo "Could not find file $file_name. Are you sure it exists?"
      file_name=""
    fi
```

```
 done

 local key
 key="$(basename "$file_name")"

 local result=0
 if copy_file_to_bucket "$bucket_name" "$file_name" "$key"; then
   echo "Uploaded file $file_name into bucket $bucket_name with key $key."
 else
   result=1
 fi

 local destination_file
 destination_file="$file_name.download"
 if yes_no_input "Would you like to download $key to the file $destination_file?
(y/n) "; then
   if download_object_from_bucket "$bucket_name" "$destination_file" "$key";
then
     echo "Downloaded $key in the bucket $bucket_name to the file
$destination_file."
   else
     result=1
   fi
 fi

 if yes_no_input "Would you like to copy $key a new object key in your bucket?
(y/n) "; then
   local to_key
   to_key="demo/$key"
   if copy_item_in_bucket "$bucket_name" "$key" "$to_key"; then
     echo "Copied $key in the bucket $bucket_name to the  $to_key."
   else
     result=1
   fi
 fi

 local bucket_items
 bucket_items=$(list_items_in_bucket "$bucket_name")

 # shellcheck disable=SC2181
 if [[ $? -ne 0 ]]; then
   result=1
 fi
```

```
  echo "Your bucket contains the following items."
  echo -e "Name\t\tSize"
  echo "$bucket_items"

  if yes_no_input "Delete the bucket, $bucket_name, as well as the objects in it?
 (y/n) "; then
    bucket_items=$(echo "$bucket_items" | cut -f 1)

    if delete_items_in_bucket "$bucket_name" "$bucket_items"; then
      echo "The following items were deleted from the bucket $bucket_name"
      echo "$bucket_items"
    else
      result=1
    fi

    if delete_bucket "$bucket_name"; then
      echo "Deleted the bucket $bucket_name"
    else
      result=1
    fi
  fi

  return $result
}
```

Die in diesem Szenario verwendeten Amazon S3-Funktionen.

```
##############################################################################
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#       -b bucket_name  -- The name of the bucket to create.
#       -r region_code  -- The code for an AWS Region in which to
#                          create the bucket.
#
# Returns:
#       The URL of the bucket that was created.
#     And:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function create_bucket() {
  local bucket_name region_code response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function create_bucket"
    echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
    echo "  -b bucket_name    The name of the bucket. It must be globally
 unique."
    echo "  [-r region_code]    The code for an AWS Region in which the bucket is
 created."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "b:r:h" option; do
    case "${option}" in
      b) bucket_name="${OPTARG}" ;;
      r) region_code="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
  fi

  local bucket_config_arg
  # A location constraint for "us-east-1" returns an error.
  if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
```

```
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
  fi

  iecho "Parameters:\n"
  iecho "    Bucket name:   $bucket_name"
  iecho "    Region code:   $region_code"
  iecho ""

  # If the bucket already exists, we don't want to try to create it.
  if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file to.
#       $2 - The path and file name of the local file to copy to the bucket.
#       $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function copy_file_to_bucket() {
  local response bucket_name source_file destination_file_name
  bucket_name=$1
```

```
  source_file=$2
  destination_file_name=$3

  response=$(aws s3api put-object \
    --bucket "$bucket_name" \
    --body "$source_file" \
    --key "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}

################################################################################
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#       $1 - The name of the bucket to download the object from.
#       $2 - The path and file name to store the downloaded bucket.
#       $3 - The key (name) of the object in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function download_object_from_bucket() {
  local bucket_name=$1
  local destination_file_name=$2
  local object_name=$3
  local response

  response=$(aws s3api get-object \
    --bucket "$bucket_name" \
    --key "$object_name" \
    "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
```

```
  fi
}

########################################################################################
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file from and to.
#       $2 - The key of the source file to copy.
#       $3 - The key of the destination file.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function copy_item_in_bucket() {
  local bucket_name=$1
  local source_key=$2
  local destination_key=$3
  local response

  response=$(aws s3api copy-object \
    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
    --key "$destination_key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api copy-object operation failed.\n$response"
    return 1
  fi
}

########################################################################################
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#       $1 - The name of the bucket.
```

```
#
# Returns:
#       The list of files in text format.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function list_items_in_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api list-objects \
    --bucket "$bucket_name" \
    --output text \
    --query 'Contents[].{Key: Key, Size: Size}')

  # shellcheck disable=SC2181
  if [[ ${?} -eq 0 ]]; then
    echo "$response"
  else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - A list of keys in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_items_in_bucket() {
  local bucket_name=$1
  local keys=$2
  local response

  # Create the JSON for the items to delete.
```

```
  local delete_items
  delete_items="{\"Objects\":["
  for key in $keys; do
    delete_items="$delete_items{\"Key\": \"$key\"},"
  done
  delete_items=${delete_items%?} # Remove the final comma.
  delete_items="$delete_items]}"

  response=$(aws s3api delete-objects \
    --bucket "$bucket_name" \
    --delete "$delete_items")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n
$response"
    return 1
  fi
}

################################################################################
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.

# Returns:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function delete_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api delete-bucket \
    --bucket "$bucket_name")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
    return 1
  fi
```

```
}
```

- API-Details finden Sie in den folgenden Themen der AWS CLI -Befehlsreferenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

C++

SDK für C++

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
#include <iostream>
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <aws/s3/model/CopyObjectRequest.h>
#include <aws/s3/model/CreateBucketRequest.h>
#include <aws/s3/model/DeleteBucketRequest.h>
#include <aws/s3/model/DeleteObjectRequest.h>
#include <aws/s3/model/GetObjectRequest.h>
#include <aws/s3/model/ListObjectsRequest.h>
#include <aws/s3/model/PutObjectRequest.h>
#include <aws/s3/model/BucketLocationConstraint.h>
#include <aws/s3/model/CreateBucketConfiguration.h>
#include <aws/core/utils/UUID.h>
#include <aws/core/utils/StringUtils.h>
#include <aws/core/utils/memory/stl/AWSAllocator.h>
#include <aws/core/utils/memory/stl/AWSStreamFwd.h>
```

```
#include <fstream>
#include "awsdoc/s3/s3_examples.h"

namespace AwsDoc {
    namespace S3 {

        //! Delete an S3 bucket.
        /*!
          \sa DeleteBucket()
          \param bucketName The S3 bucket's name.
          \param client An S3 client.
        */
        static bool DeleteBucket(const Aws::String &bucketName, Aws::S3::S3Client
 &client);

        //! Delete an object in an S3 bucket.
        /*!          \sa DeleteObjectFromBucket()
          \param bucketName The S3 bucket's name.
          \param key The key for the object in the S3 bucket.
          \param client An S3 client.
         */
        static bool
        DeleteObjectFromBucket(const Aws::String &bucketName, const Aws::String
 &key, Aws::S3::S3Client &client);
    }
}


//! Scenario to create, copy, and delete S3 buckets and objects.
/*!
  \sa S3_GettingStartedScenario()
  \param uploadFilePath Path to file to upload to an Amazon S3 bucket.
  \param saveFilePath Path for saving a downloaded S3 object.
  \param clientConfig Aws client configuration.
 */
bool AwsDoc::S3::S3_GettingStartedScenario(const Aws::String &uploadFilePath,
 const Aws::String &saveFilePath,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {

    Aws::S3::S3Client client(clientConfig);

    // Create a unique bucket name which is only temporary and will be deleted.
    // Format: "doc-example-bucket-" + lowercase UUID.
```

```
   Aws::String uuid = Aws::Utils::UUID::RandomUUID();
   Aws::String bucketName = "doc-example-bucket-" +
                            Aws::Utils::StringUtils::ToLower(uuid.c_str());

   // 1. Create a bucket.
   {
       Aws::S3::Model::CreateBucketRequest request;
       request.SetBucket(bucketName);

       if (clientConfig.region != Aws::Region::US_EAST_1) {
           Aws::S3::Model::CreateBucketConfiguration createBucketConfiguration;
           createBucketConfiguration.WithLocationConstraint(

Aws::S3::Model::BucketLocationConstraintMapper::GetBucketLocationConstraintForName(
                           clientConfig.region));
           request.WithCreateBucketConfiguration(createBucketConfiguration);
       }

       Aws::S3::Model::CreateBucketOutcome outcome =
client.CreateBucket(request);

       if (!outcome.IsSuccess()) {
           const Aws::S3::S3Error &err = outcome.GetError();
           std::cerr << "Error: CreateBucket: " <<
                     err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
           return false;
       }
       else {
           std::cout << "Created the bucket, '" << bucketName <<
                     "', in the region, '" << clientConfig.region << "'." <<
std::endl;
       }
   }

   // 2. Upload a local file to the bucket.
   Aws::String key = "key-for-test";
   {
       Aws::S3::Model::PutObjectRequest request;
       request.SetBucket(bucketName);
       request.SetKey(key);

       std::shared_ptr<Aws::FStream> input_data =
               Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
```

```
                                                uploadFilePath,
                                                std::ios_base::in |
std::ios_base::binary);

       if (!input_data->is_open()) {
           std::cerr << "Error: unable to open file, '" << uploadFilePath <<
"'." << std::endl;
           AwsDoc::S3::DeleteBucket(bucketName, client);
           return false;
       }

       request.SetBody(input_data);

       Aws::S3::Model::PutObjectOutcome outcome =
               client.PutObject(request);

       if (!outcome.IsSuccess()) {
           std::cerr << "Error: PutObject: " <<
                     outcome.GetError().GetMessage() << std::endl;
           AwsDoc::S3::DeleteObjectFromBucket(bucketName, key, client);
           AwsDoc::S3::DeleteBucket(bucketName, client);
           return false;
       }
       else {
           std::cout << "Added the object with the key, '" << key << "', to the
bucket, '"
                     << bucketName << "'." << std::endl;
       }
   }

   // 3. Download the object to a local file.
   {
       Aws::S3::Model::GetObjectRequest request;
       request.SetBucket(bucketName);
       request.SetKey(key);

       Aws::S3::Model::GetObjectOutcome outcome =
               client.GetObject(request);

       if (!outcome.IsSuccess()) {
           const Aws::S3::S3Error &err = outcome.GetError();
           std::cerr << "Error: GetObject: " <<
                     err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
```

```
        }
        else {
            std::cout << "Downloaded the object with the key, '" << key << "', in
the bucket, '"
                      << bucketName << "'." << std::endl;

            Aws::IOStream &ioStream = outcome.GetResultWithOwnership().
                    GetBody();
            Aws::OFStream outStream(saveFilePath, std::ios_base::out |
std::ios_base::binary);
            if (!outStream.is_open()) {
                std::cout << "Error: unable to open file, '" << saveFilePath <<
"'." << std::endl;
            }
            else {
                outStream << ioStream.rdbuf();
                std::cout << "Wrote the downloaded object to the file '"
                          << saveFilePath << "'." << std::endl;
            }
        }
    }

    // 4. Copy the object to a different "folder" in the bucket.
    Aws::String copiedToKey = "test-folder/" + key;
    {
        Aws::S3::Model::CopyObjectRequest request;
        request.WithBucket(bucketName)
                .WithKey(copiedToKey)
                .WithCopySource(bucketName + "/" + key);

        Aws::S3::Model::CopyObjectOutcome outcome =
                client.CopyObject(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error: CopyObject: " <<
                      outcome.GetError().GetMessage() << std::endl;
        }
        else {
            std::cout << "Copied the object with the key, '" << key << "', to the
key, '" << copiedToKey
                      << ", in the bucket, '" << bucketName << "'." << std::endl;
        }
    }

    // 5. List objects in the bucket.
```

```
    {
        Aws::S3::Model::ListObjectsRequest request;
        request.WithBucket(bucketName);

        Aws::S3::Model::ListObjectsOutcome outcome = client.ListObjects(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error: ListObjects: " <<
                      outcome.GetError().GetMessage() << std::endl;
        }
        else {
            Aws::Vector<Aws::S3::Model::Object> objects =
                    outcome.GetResult().GetContents();

            std::cout << objects.size() << " objects in the bucket, '" <<
 bucketName << "':" << std::endl;

            for (Aws::S3::Model::Object &object: objects) {
                std::cout << "     '" << object.GetKey() << "'" << std::endl;
            }
        }
    }

    // 6. Delete all objects in the bucket.
    // All objects in the bucket must be deleted before deleting the bucket.
    AwsDoc::S3::DeleteObjectFromBucket(bucketName, copiedToKey, client);
    AwsDoc::S3::DeleteObjectFromBucket(bucketName, key, client);

    // 7. Delete the bucket.
    return AwsDoc::S3::DeleteBucket(bucketName, client);
}

bool AwsDoc::S3::DeleteObjectFromBucket(const Aws::String &bucketName, const
 Aws::String &key,
                                        Aws::S3::S3Client &client) {
    Aws::S3::Model::DeleteObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(key);

    Aws::S3::Model::DeleteObjectOutcome outcome =
            client.DeleteObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: DeleteObject: " <<
```

```
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Deleted the object with the key, '" << key << "', from the
 bucket, '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

bool AwsDoc::S3::DeleteBucket(const Aws::String &bucketName, Aws::S3::S3Client
 &client) {
    Aws::S3::Model::DeleteBucketRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketOutcome outcome =
            client.DeleteBucket(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: DeleteBucket: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Deleted the bucket, '" << bucketName << "'." << std::endl;
    }
    return outcome.IsSuccess();
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for C++ -API-Referenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Definieren Sie eine Struktur, die die vom Szenario verwendete Bucket- und Objektaktionen umschließt.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// ListBuckets lists the buckets in the current account.
func (basics BucketBasics) ListBuckets() ([]types.Bucket, error) {
 result, err := basics.S3Client.ListBuckets(context.TODO(),
 &s3.ListBucketsInput{})
 var buckets []types.Bucket
 if err != nil {
  log.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
 } else {
  buckets = result.Buckets
 }
 return buckets, err
}



// BucketExists checks whether a bucket exists in the current account.
```

```
func (basics BucketBasics) BucketExists(bucketName string) (bool, error) {
 _, err := basics.S3Client.HeadBucket(context.TODO(), &s3.HeadBucketInput{
  Bucket: aws.String(bucketName),
 })
 exists := true
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NotFound:
    log.Printf("Bucket %v is available.\n", bucketName)
    exists = false
    err = nil
   default:
    log.Printf("Either you don't have access to bucket %v or another error
 occurred. "+
     "Here's what happened: %v\n", bucketName, err)
   }
  }
 } else {
  log.Printf("Bucket %v exists and you already own it.", bucketName)
 }

 return exists, err
}



// CreateBucket creates a bucket with the specified name in the specified Region.
func (basics BucketBasics) CreateBucket(name string, region string) error {
 _, err := basics.S3Client.CreateBucket(context.TODO(), &s3.CreateBucketInput{
  Bucket: aws.String(name),
  CreateBucketConfiguration: &types.CreateBucketConfiguration{
   LocationConstraint: types.BucketLocationConstraint(region),
  },
 })
 if err != nil {
  log.Printf("Couldn't create bucket %v in Region %v. Here's why: %v\n",
   name, region, err)
 }
 return err
}
```

```
// UploadFile reads from a file and puts the data into an object in a bucket.
func (basics BucketBasics) UploadFile(bucketName string, objectKey string,
 fileName string) error {
 file, err := os.Open(fileName)
 if err != nil {
  log.Printf("Couldn't open file %v to upload. Here's why: %v\n", fileName, err)
 } else {
  defer file.Close()
  _, err = basics.S3Client.PutObject(context.TODO(), &s3.PutObjectInput{
   Bucket: aws.String(bucketName),
   Key:    aws.String(objectKey),
   Body:   file,
  })
  if err != nil {
   log.Printf("Couldn't upload file %v to %v:%v. Here's why: %v\n",
    fileName, bucketName, objectKey, err)
  }
 }
 return err
}



// UploadLargeObject uses an upload manager to upload data to an object in a
 bucket.
// The upload manager breaks large data into parts and uploads the parts
 concurrently.
func (basics BucketBasics) UploadLargeObject(bucketName string, objectKey string,
 largeObject []byte) error {
 largeBuffer := bytes.NewReader(largeObject)
 var partMiBs int64 = 10
 uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
  u.PartSize = partMiBs * 1024 * 1024
 })
 _, err := uploader.Upload(context.TODO(), &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
  Body:   largeBuffer,
 })
 if err != nil {
  log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
```

```
 return err
}



// DownloadFile gets an object from a bucket and stores it in a local file.
func (basics BucketBasics) DownloadFile(bucketName string, objectKey string,
 fileName string) error {
 result, err := basics.S3Client.GetObject(context.TODO(), &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't get object %v:%v. Here's why: %v\n", bucketName,
 objectKey, err)
  return err
 }
 defer result.Body.Close()
 file, err := os.Create(fileName)
 if err != nil {
  log.Printf("Couldn't create file %v. Here's why: %v\n", fileName, err)
  return err
 }
 defer file.Close()
 body, err := io.ReadAll(result.Body)
 if err != nil {
  log.Printf("Couldn't read object body from %v. Here's why: %v\n", objectKey,
 err)
 }
 _, err = file.Write(body)
 return err
}



// DownloadLargeObject uses a download manager to download an object from a
 bucket.
// The download manager gets the data in parts and writes them to a buffer until
 all of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(bucketName string, objectKey
 string) ([]byte, error) {
 var partMiBs int64 = 10
```

```
 downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader)
 {
  d.PartSize = partMiBs * 1024 * 1024
 })
 buffer := manager.NewWriteAtBuffer([]byte{})
 _, err := downloader.Download(context.TODO(), buffer, &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return buffer.Bytes(), err
}



// CopyToFolder copies an object in a bucket to a subfolder in the same bucket.
func (basics BucketBasics) CopyToFolder(bucketName string, objectKey string,
 folderName string) error {
 _, err := basics.S3Client.CopyObject(context.TODO(), &s3.CopyObjectInput{
  Bucket:     aws.String(bucketName),
  CopySource: aws.String(fmt.Sprintf("%v/%v", bucketName, objectKey)),
  Key:        aws.String(fmt.Sprintf("%v/%v", folderName, objectKey)),
 })
 if err != nil {
  log.Printf("Couldn't copy object from %v:%v to %v:%v/%v. Here's why: %v\n",
   bucketName, objectKey, bucketName, folderName, objectKey, err)
 }
 return err
}



// CopyToBucket copies an object in a bucket to another bucket.
func (basics BucketBasics) CopyToBucket(sourceBucket string, destinationBucket
 string, objectKey string) error {
 _, err := basics.S3Client.CopyObject(context.TODO(), &s3.CopyObjectInput{
  Bucket:     aws.String(destinationBucket),
  CopySource: aws.String(fmt.Sprintf("%v/%v", sourceBucket, objectKey)),
  Key:        aws.String(objectKey),
 })
 if err != nil {
```

```
  log.Printf("Couldn't copy object from %v:%v to %v:%v. Here's why: %v\n",
   sourceBucket, objectKey, destinationBucket, objectKey, err)
 }
 return err
}



// ListObjects lists the objects in a bucket.
func (basics BucketBasics) ListObjects(bucketName string) ([]types.Object, error)
 {
 result, err := basics.S3Client.ListObjectsV2(context.TODO(),
 &s3.ListObjectsV2Input{
  Bucket: aws.String(bucketName),
 })
 var contents []types.Object
 if err != nil {
  log.Printf("Couldn't list objects in bucket %v. Here's why: %v\n", bucketName,
 err)
 } else {
  contents = result.Contents
 }
 return contents, err
}



// DeleteObjects deletes a list of objects from a bucket.
func (basics BucketBasics) DeleteObjects(bucketName string, objectKeys []string)
 error {
 var objectIds []types.ObjectIdentifier
 for _, key := range objectKeys {
  objectIds = append(objectIds, types.ObjectIdentifier{Key: aws.String(key)})
 }
 output, err := basics.S3Client.DeleteObjects(context.TODO(),
 &s3.DeleteObjectsInput{
  Bucket: aws.String(bucketName),
  Delete: &types.Delete{Objects: objectIds},
 })
 if err != nil {
  log.Printf("Couldn't delete objects from bucket %v. Here's why: %v\n",
 bucketName, err)
 } else {
  log.Printf("Deleted %v objects.\n", len(output.Deleted))
```

```
 }
 return err
}



// DeleteBucket deletes a bucket. The bucket must be empty or an error is
 returned.
func (basics BucketBasics) DeleteBucket(bucketName string) error {
 _, err := basics.S3Client.DeleteBucket(context.TODO(), &s3.DeleteBucketInput{
  Bucket: aws.String(bucketName)})
 if err != nil {
  log.Printf("Couldn't delete bucket %v. Here's why: %v\n", bucketName, err)
 }
 return err
}
```

Führen Sie ein interaktives Szenario aus, das Ihnen zeigt, wie Sie mit S3-Buckets und Objekten arbeiten.

```
// RunGetStartedScenario is an interactive example that shows you how to use
 Amazon
// Simple Storage Service (Amazon S3) to create an S3 bucket and use it to store
 objects.
//
// 1. Create a bucket.
// 2. Upload a local file to the bucket.
// 3. Upload a large object to the bucket by using an upload manager.
// 4. Download an object to a local file.
// 5. Download a large object by using a download manager.
// 6. Copy an object to a different folder in the bucket.
// 7. List objects in the bucket.
// 8. Delete all objects in the bucket.
// 9. Delete the bucket.
//
// This example creates an Amazon S3 service client from the specified sdkConfig
 so that
// you can replace it with a mocked or stubbed config for unit testing.
//
```

```
// It uses a questioner from the `demotools` package to get input during the
 example.
// This package can be found in the ..\..\demotools folder of this repo.
func RunGetStartedScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner) {
 defer func() {
  if r := recover(); r != nil {
   fmt.Println("Something went wrong with the demo.\n", r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the Amazon S3 getting started demo.")
 log.Println(strings.Repeat("-", 88))

 s3Client := s3.NewFromConfig(sdkConfig)
 bucketBasics := actions.BucketBasics{S3Client: s3Client}

 count := 10
 log.Printf("Let's list up to %v buckets for your account:", count)
 buckets, err := bucketBasics.ListBuckets()
 if err != nil {
  panic(err)
 }
 if len(buckets) == 0 {
  log.Println("You don't have any buckets!")
 } else {
  if count > len(buckets) {
   count = len(buckets)
  }
  for _, bucket := range buckets[:count] {
   log.Printf("\t%v\n", *bucket.Name)
  }
 }

 bucketName := questioner.Ask("Let's create a bucket. Enter a name for your
 bucket:",
  demotools.NotEmpty{})
 bucketExists, err := bucketBasics.BucketExists(bucketName)
 if err != nil {
  panic(err)
 }
 if !bucketExists {
  err = bucketBasics.CreateBucket(bucketName, sdkConfig.Region)
```

```
 if err != nil {
  panic(err)
 } else {
  log.Println("Bucket created.")
 }
}
log.Println(strings.Repeat("-", 88))

fmt.Println("Let's upload a file to your bucket.")
smallFile := questioner.Ask("Enter the path to a file you want to upload:",
 demotools.NotEmpty{})
const smallKey = "doc-example-key"
err = bucketBasics.UploadFile(bucketName, smallKey, smallFile)
if err != nil {
 panic(err)
}
log.Printf("Uploaded %v as %v.\n", smallFile, smallKey)
log.Println(strings.Repeat("-", 88))

mibs := 30
log.Printf("Let's create a slice of %v MiB of random bytes and upload it to your
bucket. ", mibs)
questioner.Ask("Press Enter when you're ready.")
largeBytes := make([]byte, 1024*1024*mibs)
rand.Seed(time.Now().Unix())
rand.Read(largeBytes)
largeKey := "doc-example-large"
log.Println("Uploading...")
err = bucketBasics.UploadLargeObject(bucketName, largeKey, largeBytes)
if err != nil {
 panic(err)
}
log.Printf("Uploaded %v MiB object as %v", mibs, largeKey)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's download %v to a file.", smallKey)
downloadFileName := questioner.Ask("Enter a name for the downloaded file:",
demotools.NotEmpty{})
err = bucketBasics.DownloadFile(bucketName, smallKey, downloadFileName)
if err != nil {
 panic(err)
}
log.Printf("File %v downloaded.", downloadFileName)
log.Println(strings.Repeat("-", 88))
```

```
log.Printf("Let's download the %v MiB object.", mibs)
questioner.Ask("Press Enter when you're ready.")
log.Println("Downloading...")
largeDownload, err := bucketBasics.DownloadLargeObject(bucketName, largeKey)
if err != nil {
 panic(err)
}
log.Printf("Downloaded %v bytes.", len(largeDownload))
log.Println(strings.Repeat("-", 88))

log.Printf("Let's copy %v to a folder in the same bucket.", smallKey)
folderName := questioner.Ask("Enter a folder name: ", demotools.NotEmpty{})
err = bucketBasics.CopyToFolder(bucketName, smallKey, folderName)
if err != nil {
 panic(err)
}
log.Printf("Copied %v to %v/%v.\n", smallKey, folderName, smallKey)
log.Println(strings.Repeat("-", 88))

log.Println("Let's list the objects in your bucket.")
questioner.Ask("Press Enter when you're ready.")
objects, err := bucketBasics.ListObjects(bucketName)
if err != nil {
 panic(err)
}
log.Printf("Found %v objects.\n", len(objects))
var objKeys []string
for _, object := range objects {
 objKeys = append(objKeys, *object.Key)
 log.Printf("\t%v\n", *object.Key)
}
log.Println(strings.Repeat("-", 88))

if questioner.AskBool("Do you want to delete your bucket and all of its "+
 "contents? (y/n)", "y") {
 log.Println("Deleting objects.")
 err = bucketBasics.DeleteObjects(bucketName, objKeys)
 if err != nil {
  panic(err)
 }
 log.Println("Deleting bucket.")
 err = bucketBasics.DeleteBucket(bucketName)
 if err != nil {
```

```
   panic(err)
  }
  log.Printf("Deleting downloaded file %v.\n", downloadFileName)
  err = os.Remove(downloadFileName)
  if err != nil {
   panic(err)
  }
 } else {
  log.Println("Okay. Don't forget to delete objects from your bucket to avoid
 charges.")
 }
 log.Println(strings.Repeat("-", 88))

 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Go -API-Referenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * This Java code example performs the following tasks:
 *
 * 1. Creates an Amazon S3 bucket.
 * 2. Uploads an object to the bucket.
 * 3. Downloads the object to another local file.
 * 4. Uploads an object using multipart upload.
 * 5. List all objects located in the Amazon S3 bucket.
 * 6. Copies the object to another Amazon S3 bucket.
 * 7. Deletes the object from the Amazon S3 bucket.
 * 8. Deletes the Amazon S3 bucket.
 */

public class S3Scenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException {
        final String usage = """

                Usage:
                    <bucketName> <key> <objectPath> <savePath> <toBucket>

                Where:
                    bucketName - The Amazon S3 bucket to create.
                    key - The key to use.
                    objectPath - The path where the file is located (for example,
 C:/AWS/book2.pdf).
                    savePath - The path where the file is saved after it's
 downloaded (for example, C:/AWS/book2.pdf).
                    toBucket - An Amazon S3 bucket to where an object is copied
 to (for example, C:/AWS/book2.pdf).\s
                    """;

        if (args.length != 5) {
```

```
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String key = args[1];
        String objectPath = args[2];
        String savePath = args[3];
        String toBucket = args[4];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon S3 example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create an Amazon S3 bucket.");
        createBucket(s3, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Update a local file to the Amazon S3 bucket.");
        uploadLocalFile(s3, bucketName, key, objectPath);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Download the object to another local file.");
        getObjectBytes(s3, bucketName, key, savePath);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Perform a multipart upload.");
        String multipartKey = "multiPartKey";
        multipartUpload(s3, toBucket, multipartKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. List all objects located in the Amazon S3
bucket.");
        listAllObjects(s3, bucketName);
        anotherListExample(s3, bucketName);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Copy the object to another Amazon S3 bucket.");
        copyBucketObject(s3, bucketName, key, toBucket);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Delete the object from the Amazon S3 bucket.");
        deleteObjectFromBucket(s3, bucketName, key);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Delete the Amazon S3 bucket.");
        deleteBucket(s3, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("All Amazon S3 operations were successfully
performed");
        System.out.println(DASHES);
        s3.close();
    }

    // Create a bucket by using a S3Waiter object.
    public static void createBucket(S3Client s3Client, String bucketName) {
        try {
            S3Waiter s3Waiter = s3Client.waiter();
            CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            s3Client.createBucket(bucketRequest);
            HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            // Wait until the bucket is created and print out the response.
            WaiterResponse<HeadBucketResponse> waiterResponse =
s3Waiter.waitUntilBucketExists(bucketRequestWait);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println(bucketName + " is ready");

        } catch (S3Exception e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteBucket(S3Client client, String bucket) {
        DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
                .bucket(bucket)
                .build();

        client.deleteBucket(deleteBucketRequest);
        System.out.println(bucket + " was deleted.");
    }

    /**
     * Upload an object in parts.
     */
    public static void multipartUpload(S3Client s3, String bucketName, String
key) {
        int mB = 1024 * 1024;
        // First create a multipart upload and get the upload id.
        CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

        CreateMultipartUploadResponse response =
s3.createMultipartUpload(createMultipartUploadRequest);
        String uploadId = response.uploadId();
        System.out.println(uploadId);

        // Upload all the different parts of the object.
        UploadPartRequest uploadPartRequest1 = UploadPartRequest.builder()
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)
                .partNumber(1).build();

        String etag1 = s3.uploadPart(uploadPartRequest1,
RequestBody.fromByteBuffer(getRandomByteBuffer(5 * mB)))
                .eTag();
        CompletedPart part1 =
CompletedPart.builder().partNumber(1).eTag(etag1).build();
```

```
        UploadPartRequest uploadPartRequest2 =
UploadPartRequest.builder().bucket(bucketName).key(key)
                .uploadId(uploadId)
                .partNumber(2).build();
        String etag2 = s3.uploadPart(uploadPartRequest2,
RequestBody.fromByteBuffer(getRandomByteBuffer(3 * mB)))
                .eTag();
        CompletedPart part2 =
CompletedPart.builder().partNumber(2).eTag(etag2).build();

        // Call completeMultipartUpload operation to tell S3 to merge all
uploaded
        // parts and finish the multipart operation.
        CompletedMultipartUpload completedMultipartUpload =
CompletedMultipartUpload.builder()
                .parts(part1, part2)
                .build();

        CompleteMultipartUploadRequest completeMultipartUploadRequest =
CompleteMultipartUploadRequest.builder()
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)
                .multipartUpload(completedMultipartUpload)
                .build();

        s3.completeMultipartUpload(completeMultipartUploadRequest);
    }

    private static ByteBuffer getRandomByteBuffer(int size) {
        byte[] b = new byte[size];
        new Random().nextBytes(b);
        return ByteBuffer.wrap(b);
    }

    public static void getObjectBytes(S3Client s3, String bucketName, String
keyName, String path) {
        try {
            GetObjectRequest objectRequest = GetObjectRequest
                    .builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();
```

```
            ResponseBytes<GetObjectResponse> objectBytes =
s3.getObjectAsBytes(objectRequest);
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from an S3 object");
            os.close();

        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void uploadLocalFile(S3Client s3, String bucketName, String
key, String objectPath) {
        PutObjectRequest objectRequest = PutObjectRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

        s3.putObject(objectRequest, RequestBody.fromFile(new File(objectPath)));
    }

    public static void listAllObjects(S3Client s3, String bucketName) {
        ListObjectsV2Request listObjectsReqManual =
ListObjectsV2Request.builder()
                .bucket(bucketName)
                .maxKeys(1)
                .build();

        boolean done = false;
        while (!done) {
            ListObjectsV2Response listObjResponse =
s3.listObjectsV2(listObjectsReqManual);
            for (S3Object content : listObjResponse.contents()) {
                System.out.println(content.key());
            }
```

```
            if (listObjResponse.nextContinuationToken() == null) {
                done = true;
            }

            listObjectsReqManual = listObjectsReqManual.toBuilder()
                    .continuationToken(listObjResponse.nextContinuationToken())
                    .build();
        }
    }

    public static void anotherListExample(S3Client s3, String bucketName) {
        ListObjectsV2Request listReq = ListObjectsV2Request.builder()
                .bucket(bucketName)
                .maxKeys(1)
                .build();

        ListObjectsV2Iterable listRes = s3.listObjectsV2Paginator(listReq);

        // Process response pages.
        listRes.stream()
                .flatMap(r -> r.contents().stream())
                .forEach(content -> System.out.println(" Key: " + content.key() +
" size = " + content.size()));

        // Helper method to work with paginated collection of items directly.
        listRes.contents().stream()
                .forEach(content -> System.out.println(" Key: " + content.key() +
" size = " + content.size()));

        for (S3Object content : listRes.contents()) {
            System.out.println(" Key: " + content.key() + " size = " +
content.size());
        }
    }

    public static void deleteObjectFromBucket(S3Client s3, String bucketName,
String key) {
        DeleteObjectRequest deleteObjectRequest = DeleteObjectRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

        s3.deleteObject(deleteObjectRequest);
```

```
            System.out.println(key + " was deleted");
        }

        public static String copyBucketObject(S3Client s3, String fromBucket, String
     objectKey, String toBucket) {
            String encodedUrl = null;
            try {
                encodedUrl = URLEncoder.encode(fromBucket + "/" + objectKey,
     StandardCharsets.UTF_8.toString());
            } catch (UnsupportedEncodingException e) {
                System.out.println("URL could not be encoded: " + e.getMessage());
            }
            CopyObjectRequest copyReq = CopyObjectRequest.builder()
                    .copySource(encodedUrl)
                    .destinationBucket(toBucket)
                    .destinationKey(objectKey)
                    .build();

            try {
                CopyObjectResponse copyRes = s3.copyObject(copyReq);
                System.out.println("The " + objectKey + " was copied to " +
     toBucket);
                return copyRes.copyObjectResult().toString();

            } catch (S3Exception e) {
                System.err.println(e.awsErrorDetails().errorMessage());
                System.exit(1);
            }
            return "";
        }
    }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2

- PutObject

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Importieren Sie zunächst alle erforderlichen Module.

```
// Used to check if currently running file is this file.
import { fileURLToPath } from "url";
import { readdirSync, readFileSync, writeFileSync } from "fs";

// Local helper utils.
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import { Prompter } from "@aws-doc-sdk-examples/lib/prompter.js";
import { wrapText } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

import {
  S3Client,
  CreateBucketCommand,
  PutObjectCommand,
  ListObjectsCommand,
  CopyObjectCommand,
  GetObjectCommand,
  DeleteObjectsCommand,
  DeleteBucketCommand,
} from "@aws-sdk/client-s3";
```

Die vorherigen Importe verweisen auf einige hilfreiche Dienstprogramme. Diese Dienstprogramme sind lokal im GitHub Repository, das zu Beginn dieses Abschnitts verknüpft ist. Zu Ihrer Information finden Sie im Folgenden Implementierungen dieser Dienstprogramme.

```
export const dirnameFromMetaUrl = (metaUrl) =>
  fileURLToPath(new URL(".", metaUrl));
```

```
import { select, input, confirm, checkbox } from "@inquirer/prompts";

export class Prompter {
  /**
   * @param {{ message: string, choices: { name: string, value: string }[]}}
 options
   */
  select(options) {
    return select(options);
  }

  /**
   * @param {{ message: string }} options
   */
  input(options) {
    return input(options);
  }

  /**
   * @param {string} prompt
   */
  checkContinue = async (prompt = "") => {
    const prefix = prompt && prompt + " ";
    let ok = await this.confirm({
      message: `${prefix}Continue?`,
    });
    if (!ok) throw new Error("Exiting...");
  };

  /**
   * @param {{ message: string }} options
   */
  confirm(options) {
    return confirm(options);
  }

  /**
   * @param {{ message: string, choices: { name: string, value: string }[]}}
 options
   */
  checkbox(options) {
    return checkbox(options);
  }
```

```
}

export const wrapText = (text, char = "=") => {
  const rule = char.repeat(80);
  return `${rule}\n    ${text}\n${rule}\n`;
};
```

Objekte in S3 werden in „Buckets“ gespeichert. Definieren wir eine Funktion zum Erstellen eines neuen Buckets.

```
export const createBucket = async () => {
  const bucketName = await prompter.input({
    message: "Enter a bucket name. Bucket names must be globally unique:",
  });
  const command = new CreateBucketCommand({ Bucket: bucketName });
  await s3Client.send(command);
  console.log("Bucket created successfully.\n");
  return bucketName;
};
```

Buckets enthalten „Objekte“. Diese Funktion lädt den Inhalt eines Verzeichnisses als Objekte in Ihren Bucket hoch.

```
export const uploadFilesToBucket = async ({ bucketName, folderPath }) => {
  console.log(`Uploading files from ${folderPath}\n`);
  const keys = readdirSync(folderPath);
  const files = keys.map((key) => {
    const filePath = `${folderPath}/${key}`;
    const fileContent = readFileSync(filePath);
    return {
      Key: key,
      Body: fileContent,
    };
  });

  for (let file of files) {
    await s3Client.send(
      new PutObjectCommand({
        Bucket: bucketName,
        Body: file.Body,
        Key: file.Key,
```

```
      }),
    );
    console.log(`${file.Key} uploaded successfully.`);
  }
};
```

Überprüfen Sie nach dem Hochladen von Objekten, ob sie korrekt hochgeladen wurden. Sie können ListObjects dafür verwenden. Sie werden die Eigenschaft „Key“ verwenden, die Antwort enthält jedoch noch weitere nützliche Eigenschaften.

```
export const listFilesInBucket = async ({ bucketName }) => {
  const command = new ListObjectsCommand({ Bucket: bucketName });
  const { Contents } = await s3Client.send(command);
  const contentsList = Contents.map((c) => ` • ${c.Key}`).join("\n");
  console.log("\nHere's a list of files in the bucket:");
  console.log(contentsList + "\n");
};
```

Gelegentlich möchten Sie vielleicht ein Objekt von einem Bucket in einen anderen kopieren. Verwenden Sie dafür den CopyObject Befehl .

```
export const copyFileFromBucket = async ({ destinationBucket }) => {
  const proceed = await prompter.confirm({
    message: "Would you like to copy an object from another bucket?",
  });

  if (!proceed) {
    return;
  } else {
    const copy = async () => {
      try {
        const sourceBucket = await prompter.input({
          message: "Enter source bucket name:",
        });
        const sourceKey = await prompter.input({
          message: "Enter source key:",
        });
        const destinationKey = await prompter.input({
          message: "Enter destination key:",
        });
```

```
        const command = new CopyObjectCommand({
          Bucket: destinationBucket,
          CopySource: `${sourceBucket}/${sourceKey}`,
          Key: destinationKey,
        });
        await s3Client.send(command);
        await copyFileFromBucket({ destinationBucket });
      } catch (err) {
        console.error(`Copy error.`);
        console.error(err);
        const retryAnswer = await prompter.confirm({ message: "Try again?" });
        if (retryAnswer) {
          await copy();
        }
      }
    };
    await copy();
  }
};
```

Es gibt keine SDK-Methode zum Abrufen mehrerer Objekte aus einem Bucket. Stattdessen erstellen Sie eine Liste von Objekten, die Sie herunterladen und durchlaufen können.

```
export const downloadFilesFromBucket = async ({ bucketName }) => {
  const { Contents } = await s3Client.send(
    new ListObjectsCommand({ Bucket: bucketName }),
  );
  const path = await prompter.input({
    message: "Enter destination path for files:",
  });

  for (let content of Contents) {
    const obj = await s3Client.send(
      new GetObjectCommand({ Bucket: bucketName, Key: content.Key }),
    );
    writeFileSync(
      `${path}/${content.Key}`,
      await obj.Body.transformToByteArray(),
    );
  }
  console.log("Files downloaded successfully.\n");
```

```
};
```

Es ist Zeit, Ihre Ressourcen zu bereinigen. Ein Bucket muss leer sein, bevor er gelöscht werden kann. Mit diesen beiden Funktionen leeren und löschen Sie den Bucket.

```
export const emptyBucket = async ({ bucketName }) => {
  const listObjectsCommand = new ListObjectsCommand({ Bucket: bucketName });
  const { Contents } = await s3Client.send(listObjectsCommand);
  const keys = Contents.map((c) => c.Key);

  const deleteObjectsCommand = new DeleteObjectsCommand({
    Bucket: bucketName,
    Delete: { Objects: keys.map((key) => ({ Key: key })) },
  });
  await s3Client.send(deleteObjectsCommand);
  console.log(`${bucketName} emptied successfully.\n`);
};

export const deleteBucket = async ({ bucketName }) => {
  const command = new DeleteBucketCommand({ Bucket: bucketName });
  await s3Client.send(command);
  console.log(`${bucketName} deleted successfully.\n`);
};
```

Die Funktion „main“ fasst alles zusammen. Wenn Sie diese Datei direkt ausführen, wird die Funktion „main“ aufgerufen.

```
const main = async () => {
  const OBJECT_DIRECTORY = `${dirnameFromMetaUrl(
    import.meta.url,
  )}../../../../resources/sample_files/.sample_media`;

  try {
    console.log(wrapText("Welcome to the Amazon S3 getting started example."));
    console.log("Let's create a bucket.");
    const bucketName = await createBucket();
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("File upload."));
    console.log(
```

```
      "I have some default files ready to go. You can edit the source code to
 provide your own.",
    );
    await uploadFilesToBucket({
      bucketName,
      folderPath: OBJECT_DIRECTORY,
    });

    await listFilesInBucket({ bucketName });
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("Copy files."));
    await copyFileFromBucket({ destinationBucket: bucketName });
    await listFilesInBucket({ bucketName });
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("Download files."));
    await downloadFilesFromBucket({ bucketName });

    console.log(wrapText("Clean up."));
    await emptyBucket({ bucketName });
    await deleteBucket({ bucketName });
  } catch (err) {
    console.error(err);
  }
};
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for JavaScript -API-Referenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Kotlin

SDK für Kotlin

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <bucketName> <key> <objectPath> <savePath> <toBucket>

    Where:
        bucketName - The Amazon S3 bucket to create.
        key - The key to use.
        objectPath - The path where the file is located (for example, C:/AWS/
book2.pdf).
        savePath - The path where the file is saved after it's downloaded (for
 example, C:/AWS/book2.pdf).
        toBucket - An Amazon S3 bucket to where an object is copied to (for
 example, C:/AWS/book2.pdf).
        """

    if (args.size != 4) {
        println(usage)
        exitProcess(1)
    }

    val bucketName = args[0]
    val key = args[1]
    val objectPath = args[2]
    val savePath = args[3]
    val toBucket = args[4]

    // Create an Amazon S3 bucket.
    createBucket(bucketName)

    // Update a local file to the Amazon S3 bucket.
    putObject(bucketName, key, objectPath)
```

```
    // Download the object to another local file.
    getObjectFromMrap(bucketName, key, savePath)

    // List all objects located in the Amazon S3 bucket.
    listBucketObs(bucketName)

    // Copy the object to another Amazon S3 bucket
    copyBucketOb(bucketName, key, toBucket)

    // Delete the object from the Amazon S3 bucket.
    deleteBucketObs(bucketName, key)

    // Delete the Amazon S3 bucket.
    deleteBucket(bucketName)
    println("All Amazon S3 operations were successfully performed")
}

suspend fun createBucket(bucketName: String) {
    val request = CreateBucketRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.createBucket(request)
        println("$bucketName is ready")
    }
}

suspend fun putObject(bucketName: String, objectKey: String, objectPath: String)
 {
    val metadataVal = mutableMapOf<String, String>()
    metadataVal["myVal"] = "test"

    val request = PutObjectRequest {
        bucket = bucketName
        key = objectKey
        metadata = metadataVal
        this.body = Paths.get(objectPath).asByteStream()
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        val response = s3.putObject(request)
        println("Tag information is ${response.eTag}")
```

```
    }
}

suspend fun getObjectFromMrap(bucketName: String, keyName: String, path: String)
 {
    val request = GetObjectRequest {
        key = keyName
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.getObject(request) { resp ->
            val myFile = File(path)
            resp.body?.writeToFile(myFile)
            println("Successfully read $keyName from $bucketName")
        }
    }
}

suspend fun listBucketObs(bucketName: String) {
    val request = ListObjectsRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->

        val response = s3.listObjects(request)
        response.contents?.forEach { myObject ->
            println("The name of the key is ${myObject.key}")
            println("The owner is ${myObject.owner}")
        }
    }
}

suspend fun copyBucketOb(fromBucket: String, objectKey: String, toBucket: String)
 {
    var encodedUrl = ""
    try {
        encodedUrl = URLEncoder.encode("$fromBucket/$objectKey",
 StandardCharsets.UTF_8.toString())
    } catch (e: UnsupportedEncodingException) {
        println("URL could not be encoded: " + e.message)
    }
```

```
    val request = CopyObjectRequest {
        copySource = encodedUrl
        bucket = toBucket
        key = objectKey
    }
    S3Client { region = "us-east-1" }.use { s3 ->
        s3.copyObject(request)
    }
}

suspend fun deleteBucketObs(bucketName: String, objectName: String) {
    val objectId = ObjectIdentifier {
        key = objectName
    }

    val delOb = Delete {
        objects = listOf(objectId)
    }

    val request = DeleteObjectsRequest {
        bucket = bucketName
        delete = delOb
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteObjects(request)
        println("$objectName was deleted from $bucketName")
    }
}

suspend fun deleteBucket(bucketName: String?) {
    val request = DeleteBucketRequest {
        bucket = bucketName
    }
    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteBucket(request)
        println("The $bucketName was successfully deleted!")
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Kotlin.

- CopyObject
- CreateBucket
- DeleteBucket
- DeleteObjects
- GetObject
- ListObjectsV2
- PutObject

PHP

SDK für PHP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        echo("\n");
        echo("-------------------------------------\n");
        print("Welcome to the Amazon S3 getting started demo using PHP!\n");
        echo("-------------------------------------\n");

        $region = 'us-west-2';

        $this->s3client = new S3Client([
                'region' => $region,
        ]);
        /* Inline declaration example
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);
        */

        $this->bucketName = "doc-example-bucket-" . uniqid();

        try {
            $this->s3client->createBucket([
                'Bucket' => $this->bucketName,
                'CreateBucketConfiguration' => ['LocationConstraint' => $region],
            ]);
```

```
            echo "Created bucket named: $this->bucketName \n";
        } catch (Exception $exception) {
            echo "Failed to create bucket $this->bucketName with error: " .
 $exception->getMessage();
            exit("Please fix error with bucket creation before continuing.");
        }

        $fileName = __DIR__ . "/local-file-" . uniqid();
        try {
            $this->s3client->putObject([
                'Bucket' => $this->bucketName,
                'Key' => $fileName,
                'SourceFile' => __DIR__ . '/testfile.txt'
            ]);
            echo "Uploaded $fileName to $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to upload $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with file upload before continuing.");
        }

        try {
            $file = $this->s3client->getObject([
                'Bucket' => $this->bucketName,
                'Key' => $fileName,
            ]);
            $body = $file->get('Body');
            $body->rewind();
            echo "Downloaded the file and it begins with: {$body->read(26)}.\n";
        } catch (Exception $exception) {
            echo "Failed to download $fileName from $this->bucketName with error:
 " . $exception->getMessage();
            exit("Please fix error with file downloading before continuing.");
        }

        try {
            $folder = "copied-folder";
            $this->s3client->copyObject([
                'Bucket' => $this->bucketName,
                'CopySource' => "$this->bucketName/$fileName",
                'Key' => "$folder/$fileName-copy",
            ]);
            echo "Copied $fileName to $folder/$fileName-copy.\n";
        } catch (Exception $exception) {
```

```
            echo "Failed to copy $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with object copying before continuing.");
        }

        try {
            $contents = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            echo "The contents of your bucket are: \n";
            foreach ($contents['Contents'] as $content) {
                echo $content['Key'] . "\n";
            }
        } catch (Exception $exception) {
            echo "Failed to list objects in $this->bucketName with error: " .
 $exception->getMessage();
            exit("Please fix error with listing objects before continuing.");
        }

        try {
            $objects = [];
            foreach ($contents['Contents'] as $content) {
                $objects[] = [
                    'Key' => $content['Key'],
                ];
            }
            $this->s3client->deleteObjects([
                'Bucket' => $this->bucketName,
                'Delete' => [
                    'Objects' => $objects,
                ],
            ]);
            $check = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            if (count($check) <= 0) {
                throw new Exception("Bucket wasn't empty.");
            }
            echo "Deleted all objects and folders from $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to delete $fileName from $this->bucketName with error:
 " . $exception->getMessage();
            exit("Please fix error with object deletion before continuing.");
        }
```

```
        try {
            $this->s3client->deleteBucket([
                'Bucket' => $this->bucketName,
            ]);
            echo "Deleted bucket $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to delete $this->bucketName with error: " . $exception-
>getMessage();
            exit("Please fix error with bucket deletion before continuing.");
        }

        echo "Successfully ran the Amazon S3 with PHP demo.\n";
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for PHP -API-Referenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import io
import os
import uuid
```

```
import boto3
from boto3.s3.transfer import S3UploadFailedError
from botocore.exceptions import ClientError


def do_scenario(s3_resource):
    print("-" * 88)
    print("Welcome to the Amazon S3 getting started demo!")
    print("-" * 88)

    bucket_name = f"doc-example-bucket-{uuid.uuid4()}"
    bucket = s3_resource.Bucket(bucket_name)
    try:
        bucket.create(
            CreateBucketConfiguration={
                "LocationConstraint": s3_resource.meta.client.meta.region_name
            }
        )
        print(f"Created demo bucket named {bucket.name}.")
    except ClientError as err:
        print(f"Tried and failed to create demo bucket {bucket_name}.")
        print(f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}")
        print(f"\nCan't continue the demo without a bucket!")
        return

    file_name = None
    while file_name is None:
        file_name = input("\nEnter a file you want to upload to your bucket: ")
        if not os.path.exists(file_name):
            print(f"Couldn't find file {file_name}. Are you sure it exists?")
            file_name = None

    obj = bucket.Object(os.path.basename(file_name))
    try:
        obj.upload_file(file_name)
        print(
            f"Uploaded file {file_name} into bucket {bucket.name} with key
 {obj.key}."
        )
    except S3UploadFailedError as err:
        print(f"Couldn't upload file {file_name} to {bucket.name}.")
        print(f"\t{err}")
```

```
    answer = input(f"\nDo you want to download {obj.key} into memory (y/n)? ")
    if answer.lower() == "y":
        data = io.BytesIO()
        try:
            obj.download_fileobj(data)
            data.seek(0)
            print(f"Got your object. Here are the first 20 bytes:\n")
            print(f"\t{data.read(20)}")
        except ClientError as err:
            print(f"Couldn't download {obj.key}.")
            print(
                f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}"
            )

    answer = input(
        f"\nDo you want to copy {obj.key} to a subfolder in your bucket (y/n)? "
    )
    if answer.lower() == "y":
        dest_obj = bucket.Object(f"demo-folder/{obj.key}")
        try:
            dest_obj.copy({"Bucket": bucket.name, "Key": obj.key})
            print(f"Copied {obj.key} to {dest_obj.key}.")
        except ClientError as err:
            print(f"Couldn't copy {obj.key} to {dest_obj.key}.")
            print(
                f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}"
            )

    print("\nYour bucket contains the following objects:")
    try:
        for o in bucket.objects.all():
            print(f"\t{o.key}")
    except ClientError as err:
        print(f"Couldn't list the objects in bucket {bucket.name}.")
        print(f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}")

    answer = input(
        "\nDo you want to delete all of the objects as well as the bucket (y/n)?
 "
    )
```

```
    if answer.lower() == "y":
        try:
            bucket.objects.delete()
            bucket.delete()
            print(f"Emptied and deleted bucket {bucket.name}.\n")
        except ClientError as err:
            print(f"Couldn't empty and delete bucket {bucket.name}.")
            print(
                f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}"
            )

    print("Thanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
    do_scenario(boto3.resource("s3"))
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Ruby

SDK für Ruby

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
require "aws-sdk-s3"

# Wraps the getting started scenario actions.
class ScenarioGettingStarted
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
  end

  # Creates a bucket with a random name in the currently configured account and
  # AWS Region.
  #
  # @return [Aws::S3::Bucket] The newly created bucket.
  def create_bucket
    bucket = @s3_resource.create_bucket(
      bucket: "doc-example-bucket-#{Random.uuid}",
      create_bucket_configuration: {
        location_constraint: "us-east-1" # Note: only certain regions permitted
      }
    )
    puts("Created demo bucket named #{bucket.name}.")
  rescue Aws::Errors::ServiceError => e
    puts("Tried and failed to create demo bucket.")
    puts("\t#{e.code}: #{e.message}")
    puts("\nCan't continue the demo without a bucket!")
    raise
  else
    bucket
  end

  # Requests a file name from the user.
  #
  # @return The name of the file.
  def create_file
    File.open("demo.txt", w) { |f| f.write("This is a demo file.") }
  end

  # Uploads a file to an Amazon S3 bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket object representing the upload
 destination
```

```
 # @return [Aws::S3::Object] The Amazon S3 object that contains the uploaded
file.
 def upload_file(bucket)
   File.open("demo.txt", "w+") { |f| f.write("This is a demo file.") }
   s3_object = bucket.object(File.basename("demo.txt"))
   s3_object.upload_file("demo.txt")
   puts("Uploaded file demo.txt into bucket #{bucket.name} with key
#{s3_object.key}.")
 rescue Aws::Errors::ServiceError => e
   puts("Couldn't upload file demo.txt to #{bucket.name}.")
   puts("\t#{e.code}: #{e.message}")
   raise
 else
   s3_object
 end

 # Downloads an Amazon S3 object to a file.
 #
 # @param s3_object [Aws::S3::Object] The object to download.
 def download_file(s3_object)
   puts("\nDo you want to download #{s3_object.key} to a local file (y/n)? ")
   answer = gets.chomp.downcase
   if answer == "y"
     puts("Enter a name for the downloaded file: ")
     file_name = gets.chomp
     s3_object.download_file(file_name)
     puts("Object #{s3_object.key} successfully downloaded to #{file_name}.")
   end
 rescue Aws::Errors::ServiceError => e
   puts("Couldn't download #{s3_object.key}.")
   puts("\t#{e.code}: #{e.message}")
   raise
 end

 # Copies an Amazon S3 object to a subfolder within the same bucket.
 #
 # @param source_object [Aws::S3::Object] The source object to copy.
 # @return [Aws::S3::Object, nil] The destination object.
 def copy_object(source_object)
   dest_object = nil
   puts("\nDo you want to copy #{source_object.key} to a subfolder in your
bucket (y/n)? ")
   answer = gets.chomp.downcase
   if answer == "y"
```

```
      dest_object = source_object.bucket.object("demo-folder/
#{source_object.key}")
      dest_object.copy_from(source_object)
      puts("Copied #{source_object.key} to #{dest_object.key}.")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't copy #{source_object.key}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    dest_object
  end

  # Lists the objects in an Amazon S3 bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket to query.
  def list_objects(bucket)
    puts("\nYour bucket contains the following objects:")
    bucket.objects.each do |obj|
      puts("\t#{obj.key}")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't list the objects in bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes the objects in an Amazon S3 bucket and deletes the bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
  def delete_bucket(bucket)
    puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?
 ")
    answer = gets.chomp.downcase
    if answer == "y"
      bucket.objects.batch_delete!
      bucket.delete
      puts("Emptied and deleted bucket #{bucket.name}.\n")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't empty and delete bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
```

```
end

# Runs the Amazon S3 getting started scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the Amazon S3 getting started demo!")
  puts("-" * 88)

  bucket = scenario.create_bucket
  s3_object = scenario.upload_file(bucket)
  scenario.download_file(s3_object)
  scenario.copy_object(s3_object)
  scenario.list_objects(bucket)
  scenario.delete_bucket(bucket)

  puts("Thanks for watching!")
  puts("-" * 88)
rescue Aws::Errors::ServiceError
  puts("Something went wrong with the demo!")
end

run_scenario(ScenarioGettingStarted.new(Aws::S3::Resource.new)) if $PROGRAM_NAME
 == __FILE__
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Ruby -API-Referenz.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Code für die binäre Crate-Datei, die das Szenario ausführt.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_s3::{config::Region, Client};
use s3_service::error::Error;
use uuid::Uuid;


#[tokio::main]
async fn main() -> Result<(), Error> {
    let (region, client, bucket_name, file_name, key, target_key) =
 initialize_variables().await;

    if let Err(e) = run_s3_operations(region, client, bucket_name, file_name,
 key, target_key).await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (Region, Client, String, String, String,
 String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));
    let region = region_provider.region().await.unwrap();

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = Client::new(&shared_config);
```

```
    let bucket_name = format!("doc-example-bucket-{}", Uuid::new_v4());

    let file_name = "s3/testfile.txt".to_string();
    let key = "test file key name".to_string();
    let target_key = "target_key".to_string();

    (region, client, bucket_name, file_name, key, target_key)
}

async fn run_s3_operations(
    region: Region,
    client: Client,
    bucket_name: String,
    file_name: String,
    key: String,
    target_key: String,
) -> Result<(), Error> {
    s3_service::create_bucket(&client, &bucket_name, region.as_ref()).await?;
    s3_service::upload_object(&client, &bucket_name, &file_name, &key).await?;
    let _object = s3_service::download_object(&client, &bucket_name, &key).await;
    s3_service::copy_object(&client, &bucket_name, &key, &target_key).await?;
    s3_service::list_objects(&client, &bucket_name).await?;
    s3_service::delete_objects(&client, &bucket_name).await?;
    s3_service::delete_bucket(&client, &bucket_name).await?;

    Ok(())
}
```

Eine Bibliotheks-Crate-Datei mit gemeinsamen Aktionen, die von der binären Datei aufgerufen werden.

```
use aws_sdk_s3::operation::{
    copy_object::{CopyObjectError, CopyObjectOutput},
    create_bucket::{CreateBucketError, CreateBucketOutput},
    get_object::{GetObjectError, GetObjectOutput},
    list_objects_v2::ListObjectsV2Output,
    put_object::{PutObjectError, PutObjectOutput},
};
use aws_sdk_s3::types::{
```

```
    BucketLocationConstraint, CreateBucketConfiguration, Delete,
 ObjectIdentifier,
};
use aws_sdk_s3::{error::SdkError, primitives::ByteStream, Client};
use error::Error;
use std::path::Path;
use std::str;

pub mod error;

pub async fn delete_bucket(client: &Client, bucket_name: &str) -> Result<(),
 Error> {
    client.delete_bucket().bucket(bucket_name).send().await?;
    println!("Bucket deleted");
    Ok(())
}

pub async fn delete_objects(client: &Client, bucket_name: &str) ->
 Result<Vec<String>, Error> {
    let objects = client.list_objects_v2().bucket(bucket_name).send().await?;

    let mut delete_objects: Vec<ObjectIdentifier> = vec![];
    for obj in objects.contents() {
        let obj_id = ObjectIdentifier::builder()
            .set_key(Some(obj.key().unwrap().to_string()))
            .build()
            .map_err(Error::from)?;
        delete_objects.push(obj_id);
    }

    let return_keys = delete_objects.iter().map(|o| o.key.clone()).collect();

    if !delete_objects.is_empty() {
        client
            .delete_objects()
            .bucket(bucket_name)
            .delete(
                Delete::builder()
                    .set_objects(Some(delete_objects))
                    .build()
                    .map_err(Error::from)?,
            )
            .send()
            .await?;
```

```
    }

    let objects: ListObjectsV2Output =
 client.list_objects_v2().bucket(bucket_name).send().await?;

    eprintln!("{objects:?}");

    match objects.key_count {
        Some(0) => Ok(return_keys),
        _ => Err(Error::unhandled(
            "There were still objects left in the bucket.",
        )),
    }
}

pub async fn list_objects(client: &Client, bucket: &str) -> Result<(), Error> {
    let mut response = client
        .list_objects_v2()
        .bucket(bucket.to_owned())
        .max_keys(10) // In this example, go 10 at a time.
        .into_paginator()
        .send();

    while let Some(result) = response.next().await {
        match result {
            Ok(output) => {
                for object in output.contents() {
                    println!(" - {}", object.key().unwrap_or("Unknown"));
                }
            }
            Err(err) => {
                eprintln!("{err:?}")
            }
        }
    }

    Ok(())
}

pub async fn copy_object(
    client: &Client,
    bucket_name: &str,
    object_key: &str,
    target_key: &str,
```

```
) -> Result<CopyObjectOutput, SdkError<CopyObjectError>> {
    let mut source_bucket_and_object: String = "".to_owned();
    source_bucket_and_object.push_str(bucket_name);
    source_bucket_and_object.push('/');
    source_bucket_and_object.push_str(object_key);

    client
        .copy_object()
        .copy_source(source_bucket_and_object)
        .bucket(bucket_name)
        .key(target_key)
        .send()
        .await
}

pub async fn download_object(
    client: &Client,
    bucket_name: &str,
    key: &str,
) -> Result<GetObjectOutput, SdkError<GetObjectError>> {
    client
        .get_object()
        .bucket(bucket_name)
        .key(key)
        .send()
        .await
}

pub async fn upload_object(
    client: &Client,
    bucket_name: &str,
    file_name: &str,
    key: &str,
) -> Result<PutObjectOutput, SdkError<PutObjectError>> {
    let body = ByteStream::from_path(Path::new(file_name)).await;
    client
        .put_object()
        .bucket(bucket_name)
        .key(key)
        .body(body.unwrap())
        .send()
        .await
}
```

```
pub async fn create_bucket(
    client: &Client,
    bucket_name: &str,
    region: &str,
) -> Result<CreateBucketOutput, SdkError<CreateBucketError>> {
    let constraint = BucketLocationConstraint::from(region);
    let cfg = CreateBucketConfiguration::builder()
        .location_constraint(constraint)
        .build();
    client
        .create_bucket()
        .create_bucket_configuration(cfg)
        .bucket(bucket_name)
        .send()
        .await
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zu AWS - SDK für Rust.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
DATA(lo_session) = /aws1/cl_rt_session_aws=>create( cv_pfl ).
DATA(lo_s3) = /aws1/cl_s3_factory=>create( lo_session ).

" Create an Amazon Simple Storage Service (Amazon S3) bucket. "
TRY.
    lo_s3->createbucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'S3 bucket created.' TYPE 'I'.
  CATCH /aws1/cx_s3_bucketalrdyexists.
    MESSAGE 'Bucket name already exists.' TYPE 'E'.
  CATCH /aws1/cx_s3_bktalrdyownedbyyou.
    MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.


"Upload an object to an S3 bucket."
TRY.
    "Get contents of file from application server."
    DATA lv_file_content TYPE xstring.
    OPEN DATASET iv_key FOR INPUT IN BINARY MODE.
    READ DATASET iv_key INTO lv_file_content.
    CLOSE DATASET iv_key.

    lo_s3->putobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_key
        iv_body = lv_file_content
    ).
    MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

" Get an object from a bucket. "
TRY.
    DATA(lo_result) = lo_s3->getobject(
                iv_bucket = iv_bucket_name
                iv_key = iv_key
            ).
    DATA(lv_object_data) = lo_result->get_body( ).
    MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
```

```
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
  CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

" Copy an object to a subfolder in a bucket. "
TRY.
    lo_s3->copyobject(
      iv_bucket = iv_bucket_name
      iv_key = |{ iv_copy_to_folder }/{ iv_key }|
      iv_copysource = |{ iv_bucket_name }/{ iv_key }|
    ).
    MESSAGE 'Object copied to a subfolder.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
  CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

" List objects in the bucket. "
TRY.
    DATA(lo_list) = lo_s3->listobjects(
       iv_bucket = iv_bucket_name
     ).
    MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
DATA text TYPE string VALUE 'Object List - '.
DATA lv_object_key TYPE /aws1/s3_objectkey.
LOOP AT lo_list->get_contents( ) INTO DATA(lo_object).
  lv_object_key = lo_object->get_key( ).
  CONCATENATE lv_object_key ', ' INTO text.
ENDLOOP.
MESSAGE text TYPE'I'.

" Delete the objects in a bucket. "
TRY.
    lo_s3->deleteobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_key
    ).
    lo_s3->deleteobject(
        iv_bucket = iv_bucket_name
```

```
            iv_key = |{ iv_copy_to_folder }/{ iv_key }|
        ).
        MESSAGE 'Objects deleted from S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.


    " Delete the bucket. "
    TRY.
        lo_s3->deletebucket(
            iv_bucket = iv_bucket_name
        ).
        MESSAGE 'Deleted S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS SDK für SAP ABAP.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Eine Swift-Klasse, die Aufrufe des SDK für Swift verarbeitet

```
import Foundation
import AWSS3
import ClientRuntime
import AWSClientRuntime

/// A class containing all the code that interacts with the AWS SDK for Swift.
public class ServiceHandler {
    let client: S3Client

    /// Initialize and return a new ``ServiceHandler`` object, which is used to
 drive the AWS calls
    /// used for the example.
    ///
    /// - Returns: A new ``ServiceHandler`` object, ready to be called to
    ///            execute AWS operations.
    public init() async {
        do {
            client = try S3Client(region: "us-east-2")
        } catch {
            print("ERROR: ", dump(error, name: "Initializing S3 client"))
            exit(1)
        }
    }

    /// Create a new user given the specified name.
    ///
    /// - Parameters:
    ///   - name: Name of the bucket to create.
    /// Throws an exception if an error occurs.
    public func createBucket(name: String) async throws {
        let config = S3ClientTypes.CreateBucketConfiguration(
            locationConstraint: .usEast2
        )
        let input = CreateBucketInput(
            bucket: name,
```

```
            createBucketConfiguration: config
        )
        _ = try await client.createBucket(input: input)
    }

    /// Delete a bucket.
    /// - Parameter name: Name of the bucket to delete.
    public func deleteBucket(name: String) async throws {
        let input = DeleteBucketInput(
            bucket: name
        )
        _ = try await client.deleteBucket(input: input)
    }

    /// Upload a file from local storage to the bucket.
    /// - Parameters:
    ///   - bucket: Name of the bucket to upload the file to.
    ///   - key: Name of the file to create.
    ///   - file: Path name of the file to upload.
    public func uploadFile(bucket: String, key: String, file: String) async
throws {
        let fileUrl = URL(fileURLWithPath: file)
        let fileData = try Data(contentsOf: fileUrl)
        let dataStream = ByteStream.from(data: fileData)

        let input = PutObjectInput(
            body: dataStream,
            bucket: bucket,
            key: key
        )
        _ = try await client.putObject(input: input)
    }

    /// Create a file in the specified bucket with the given name. The new
    /// file's contents are uploaded from a `Data` object.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket to create a file in.
    ///   - key: Name of the file to create.
    ///   - data: A `Data` object to write into the new file.
    public func createFile(bucket: String, key: String, withData data: Data)
async throws {
        let dataStream = ByteStream.from(data: data)
```

```
        let input = PutObjectInput(
            body: dataStream,
            bucket: bucket,
            key: key
        )
        _ = try await client.putObject(input: input)
    }

    /// Download the named file to the given directory on the local device.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket that contains the file to be copied.
    ///   - key: The name of the file to copy from the bucket.
    ///   - to: The path of the directory on the local device where you want to
    ///     download the file.
    public func downloadFile(bucket: String, key: String, to: String) async
throws {
        let fileUrl = URL(fileURLWithPath: to).appendingPathComponent(key)

        let input = GetObjectInput(
            bucket: bucket,
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the data stream object. Return immediately if there isn't one.
        guard let body = output.body,
              let data = try await body.readData() else {
            return
        }
        try data.write(to: fileUrl)
    }

    /// Read the specified file from the given S3 bucket into a Swift
    /// `Data` object.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket containing the file to read.
    ///   - key: Name of the file within the bucket to read.
    ///
    /// - Returns: A `Data` object containing the complete file data.
    public func readFile(bucket: String, key: String) async throws -> Data {
        let input = GetObjectInput(
            bucket: bucket,
```

```
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the stream and return its contents in a `Data` object. If
        // there is no stream, return an empty `Data` object instead.
        guard let body = output.body,
              let data = try await body.readData() else {
            return "".data(using: .utf8)!
        }

        return data
    }

    /// Copy a file from one bucket to another.
    ///
    /// - Parameters:
    ///   - sourceBucket: Name of the bucket containing the source file.
    ///   - name: Name of the source file.
    ///   - destBucket: Name of the bucket to copy the file into.
    public func copyFile(from sourceBucket: String, name: String, to destBucket:
 String) async throws {
        let srcUrl = ("\(sourceBucket)/
\(name)").addingPercentEncoding(withAllowedCharacters: .urlPathAllowed)

        let input = CopyObjectInput(
            bucket: destBucket,
            copySource: srcUrl,
            key: name
        )
        _ = try await client.copyObject(input: input)
    }

    /// Deletes the specified file from Amazon S3.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket containing the file to delete.
    ///   - key: Name of the file to delete.
    ///
    public func deleteFile(bucket: String, key: String) async throws {
        let input = DeleteObjectInput(
            bucket: bucket,
            key: key
        )
```

```
        do {
            _ = try await client.deleteObject(input: input)
        } catch {
            throw error
        }
    }

    /// Returns an array of strings, each naming one file in the
    /// specified bucket.
    ///
    /// - Parameter bucket: Name of the bucket to get a file listing for.
    /// - Returns: An array of `String` objects, each giving the name of
    ///            one file contained in the bucket.
    public func listBucketFiles(bucket: String) async throws -> [String] {
        let input = ListObjectsV2Input(
            bucket: bucket
        )
        let output = try await client.listObjectsV2(input: input)
        var names: [String] = []

        guard let objList = output.contents else {
            return []
        }

        for obj in objList {
            if let objName = obj.key {
                names.append(objName)
            }
        }

        return names
    }
}
```

Ein Swift-Befehlszeilenprogramm zur Verwaltung der SDK-Aufrufe

```
import Foundation
import ServiceHandler
import ArgumentParser

/// The command-line arguments and options available for this
```

```
/// example command.
struct ExampleCommand: ParsableCommand {
    @Argument(help: "Name of the S3 bucket to create")
    var bucketName: String

    @Argument(help: "Pathname of the file to upload to the S3 bucket")
    var uploadSource: String

    @Argument(help: "The name (key) to give the file in the S3 bucket")
    var objName: String

    @Argument(help: "S3 bucket to copy the object to")
    var destBucket: String

    @Argument(help: "Directory where you want to download the file from the S3
 bucket")
    var downloadDir: String

    static var configuration = CommandConfiguration(
        commandName: "s3-basics",
        abstract: "Demonstrates a series of basic AWS S3 functions.",
        discussion: """
        Performs the following Amazon S3 commands:

        * `CreateBucket`
        * `PutObject`
        * `GetObject`
        * `CopyObject`
        * `ListObjects`
        * `DeleteObjects`
        * `DeleteBucket`
        """
    )

    /// Called by ``main()`` to do the actual running of the AWS
    /// example.
    func runAsync() async throws {
        let serviceHandler = await ServiceHandler()

        // 1. Create the bucket.
        print("Creating the bucket \(bucketName)...")
        try await serviceHandler.createBucket(name: bucketName)

        // 2. Upload a file to the bucket.
```

```
        print("Uploading the file \(uploadSource)...")
        try await serviceHandler.uploadFile(bucket: bucketName, key: objName,
 file: uploadSource)

        // 3. Download the file.
        print("Downloading the file \(objName) to \(downloadDir)...")
        try await serviceHandler.downloadFile(bucket: bucketName, key: objName,
 to: downloadDir)

        // 4. Copy the file to another bucket.
        print("Copying the file to the bucket \(destBucket)...")
        try await serviceHandler.copyFile(from: bucketName, name: objName, to:
 destBucket)

        // 5. List the contents of the bucket.

        print("Getting a list of the files in the bucket \(bucketName)")
        let fileList = try await serviceHandler.listBucketFiles(bucket:
 bucketName)
        let numFiles = fileList.count
        if numFiles != 0 {
            print("\(numFiles) file\((numFiles > 1) ? "s" : "") in bucket
 \(bucketName):")
            for name in fileList {
                print("  \(name)")
            }
        } else {
            print("No files found in bucket \(bucketName)")
        }

        // 6. Delete the objects from the bucket.

        print("Deleting the file \(objName) from the bucket \(bucketName)...")
        try await serviceHandler.deleteFile(bucket: bucketName, key: objName)
        print("Deleting the file \(objName) from the bucket \(destBucket)...")
        try await serviceHandler.deleteFile(bucket: destBucket, key: objName)

        // 7. Delete the bucket.
        print("Deleting the bucket \(bucketName)...")
        try await serviceHandler.deleteBucket(name: bucketName)

        print("Done.")
    }
}
```

```
//
// Main program entry point.
//
@main
struct Main {
    static func main() async {
        let args = Array(CommandLine.arguments.dropFirst())

        do {
            let command = try ExampleCommand.parse(args)
            try await command.runAsync()
        } catch {
            ExampleCommand.exit(withError: error)
        }
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS SDK für Swift.
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte mit der Verschlüsselung für Amazon S3-Objekte mithilfe eines - AWS SDK

Das folgende Codebeispiel veranschaulicht die ersten Schritte mit der Verschlüsselung von Amazon-S3-Objekten.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to apply client encryption to an object in an
/// Amazon Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class SSEClientEncryption
{
    public static async Task Main()
    {
        string bucketName = "doc-example-bucket";
        string keyName = "exampleobject.txt";
        string copyTargetKeyName = "examplecopy.txt";

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USWest2.
        IAmazonS3 client = new AmazonS3Client();

        try
        {
            // Create an encryption key.
            Aes aesEncryption = Aes.Create();
            aesEncryption.KeySize = 256;
            aesEncryption.GenerateKey();
            string base64Key = Convert.ToBase64String(aesEncryption.Key);
```

```
                // Upload the object.
                PutObjectRequest putObjectRequest = await
UploadObjectAsync(client, bucketName, keyName, base64Key);

                // Download the object and verify that its contents match what
you uploaded.
                await DownloadObjectAsync(client, bucketName, keyName, base64Key,
putObjectRequest);

                // Get object metadata and verify that the object uses AES-256
encryption.
                await GetObjectMetadataAsync(client, bucketName, keyName,
base64Key);

                // Copy both the source and target objects using server-side
encryption with
                // an encryption key.
                await CopyObjectAsync(client, bucketName, keyName,
copyTargetKeyName, aesEncryption, base64Key);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: {ex.Message}");
            }
        }

        /// <summary>
        /// Uploads an object to an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// PutObjectAsync.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket to which
the
        /// object will be uploaded.</param>
        /// <param name="keyName">The name of the object to upload to the Amazon
S3
        /// bucket.</param>
        /// <param name="base64Key">The encryption key.</param>
        /// <returns>The PutObjectRequest object for use by
DownloadObjectAsync.</returns>
        public static async Task<PutObjectRequest> UploadObjectAsync(
            IAmazonS3 client,
```

```
            string bucketName,
            string keyName,
            string base64Key)
        {
            PutObjectRequest putObjectRequest = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
                ContentBody = "sample text",
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };
            PutObjectResponse putObjectResponse = await
 client.PutObjectAsync(putObjectRequest);
            return putObjectRequest;
        }

        /// <summary>
        /// Downloads an encrypted object from an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// GetObjectAsync.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
 object
        /// is located.</param>
        /// <param name="keyName">The name of the Amazon S3 object to download.</
param>
        /// <param name="base64Key">The encryption key used to encrypt the
        /// object.</param>
        /// <param name="putObjectRequest">The PutObjectRequest used to upload
        /// the object.</param>
        public static async Task DownloadObjectAsync(
            IAmazonS3 client,
            string bucketName,
            string keyName,
            string base64Key,
            PutObjectRequest putObjectRequest)
        {
            GetObjectRequest getObjectRequest = new GetObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
```

```
                // Provide encryption information for the object stored in Amazon
S3.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            using (GetObjectResponse getResponse = await
client.GetObjectAsync(getObjectRequest))
            using (StreamReader reader = new
StreamReader(getResponse.ResponseStream))
            {
                string content = reader.ReadToEnd();
                if (string.Compare(putObjectRequest.ContentBody, content) == 0)
                {
                    Console.WriteLine("Object content is same as we uploaded");
                }
                else
                {
                    Console.WriteLine("Error...Object content is not same.");
                }

                if (getResponse.ServerSideEncryptionCustomerMethod ==
ServerSideEncryptionCustomerMethod.AES256)
                {
                    Console.WriteLine("Object encryption method is AES256, same
as we set");
                }
                else
                {
                    Console.WriteLine("Error...Object encryption method is not
the same as AES256 we set");
                }
            }
        }

        /// <summary>
        /// Retrieves the metadata associated with an Amazon S3 object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to call GetObjectMetadataAsync.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket containing
the
```

```
        /// object for which we want to retrieve metadata.</param>
        /// <param name="keyName">The name of the object for which we wish to
        /// retrieve the metadata.</param>
        /// <param name="base64Key">The encryption key associated with the
        /// object.</param>
        public static async Task GetObjectMetadataAsync(
            IAmazonS3 client,
            string bucketName,
            string keyName,
            string base64Key)
        {
            GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest
            {
                BucketName = bucketName,
                Key = keyName,

                // The object stored in Amazon S3 is encrypted, so provide the
necessary encryption information.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            GetObjectMetadataResponse getObjectMetadataResponse = await
client.GetObjectMetadataAsync(getObjectMetadataRequest);
            Console.WriteLine("The object metadata show encryption method used
is: {0}", getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
        }

        /// <summary>
        /// Copies an encrypted object from one Amazon S3 bucket to another.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// CopyObjectAsync.</param>
        /// <param name="bucketName">The Amazon S3 bucket containing the object
        /// to copy.</param>
        /// <param name="keyName">The name of the object to copy.</param>
        /// <param name="copyTargetKeyName">The Amazon S3 bucket to which the
object
        /// will be copied.</param>
        /// <param name="aesEncryption">The encryption type to use.</param>
        /// <param name="base64Key">The encryption key to use.</param>
```

```
        public static async Task CopyObjectAsync(
            IAmazonS3 client,
            string bucketName,
            string keyName,
            string copyTargetKeyName,
            Aes aesEncryption,
            string base64Key)
        {
            aesEncryption.GenerateKey();
            string copyBase64Key = Convert.ToBase64String(aesEncryption.Key);

            CopyObjectRequest copyRequest = new CopyObjectRequest
            {
                SourceBucket = bucketName,
                SourceKey = keyName,
                DestinationBucket = bucketName,
                DestinationKey = copyTargetKeyName,

                // Information about the source object's encryption.
                CopySourceServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                CopySourceServerSideEncryptionCustomerProvidedKey = base64Key,

                // Information about the target object's encryption.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = copyBase64Key,
            };
            await client.CopyObjectAsync(copyRequest);
        }
    }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - CopyObject
  - GetObject
  - GetObjectMetadata

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte mit Tags für Amazon S3-Objekte unter Verwendung eines - AWS SDK

Das folgende Codebeispiel zeigt die ersten Schritte mit Tags für Amazon-S3-Objekte.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to work with tags in Amazon Simple Storage
/// Service (Amazon S3) objects.
/// </summary>
public class ObjectTag
{
    public static async Task Main()
    {
        string bucketName = "doc-example-bucket";
        string keyName = "newobject.txt";
        string filePath = @"*** file path ***";

        // Specify your bucket region (an example region is shown).
        RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
```

```
            var client = new AmazonS3Client(bucketRegion);
            await PutObjectsWithTagsAsync(client, bucketName, keyName, filePath);
        }

        /// <summary>
        /// This method uploads an object with tags. It then shows the tag
        /// values, changes the tags, and shows the new tags.
        /// </summary>
        /// <param name="client">The Initialized Amazon S3 client object used
        /// to call the methods to create and change an objects tags.</param>
        /// <param name="bucketName">A string representing the name of the
        /// bucket where the object will be stored.</param>
        /// <param name="keyName">A string representing the key name of the
        /// object to be tagged.</param>
        /// <param name="filePath">The directory location and file name of the
        /// object to be uploaded to the Amazon S3 bucket.</param>
        public static async Task PutObjectsWithTagsAsync(IAmazonS3 client, string
bucketName, string keyName, string filePath)
        {
            try
            {
                // Create an object with tags.
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    FilePath = filePath,
                    TagSet = new List<Tag>
                    {
                        new Tag { Key = "Keyx1", Value = "Value1" },
                        new Tag { Key = "Keyx2", Value = "Value2" },
                    },
                };

                PutObjectResponse response = await
client.PutObjectAsync(putRequest);

                // Now retrieve the new object's tags.
                GetObjectTaggingRequest getTagsRequest = new
GetObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                };
```

```
                GetObjectTaggingResponse objectTags = await
client.GetObjectTaggingAsync(getTagsRequest);

                // Display the tag values.
                objectTags.Tagging
                    .ForEach(t => Console.WriteLine($"Key: {t.Key}, Value:
{t.Value}"));

                Tagging newTagSet = new Tagging()
                {
                    TagSet = new List<Tag>
                    {
                        new Tag { Key = "Key3", Value = "Value3" },
                        new Tag { Key = "Key4", Value = "Value4" },
                    },
                };

                PutObjectTaggingRequest putObjTagsRequest = new
PutObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                    Tagging = newTagSet,
                };

                PutObjectTaggingResponse response2 = await
client.PutObjectTaggingAsync(putObjTagsRequest);

                // Retrieve the tags again and show the values.
                GetObjectTaggingRequest getTagsRequest2 = new
GetObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                };
                GetObjectTaggingResponse objectTags2 = await
client.GetObjectTaggingAsync(getTagsRequest2);

                objectTags2.Tagging
                    .ForEach(t => Console.WriteLine($"Key: {t.Key}, Value:
{t.Value}"));
            }
            catch (AmazonS3Exception ex)
```

```
            {
                Console.WriteLine(
                        $"Error: '{ex.Message}'");
            }
        }
    }
```

- Weitere API-Informationen finden Sie unter GetObjectTagging in der APIAWS SDK for .NET -Referenz für .

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Arbeiten mit Amazon S3-Objektsperrfunktionen unter Verwendung eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie mit S3-Objektsperrfunktionen arbeiten.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario aus, das die Features der Amazon S3-Objektsperre demonstriert.

```
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
```

```
using Microsoft.Extensions.Logging;
using Microsoft.Extensions.Logging.Console;
using Microsoft.Extensions.Logging.Debug;

namespace S3ObjectLockScenario;

public static class S3ObjectLockWorkflow
{
    /*
    Before running this .NET code example, set up your development environment,
 including your credentials.

    This .NET example performs the following tasks:
        1. Create test Amazon Simple Storage Service (S3) buckets with different
 lock policies.
        2. Upload sample objects to each bucket.
        3. Set some Legal Hold and Retention Periods on objects and buckets.
        4. Investigate lock policies by viewing settings or attempting to delete
 or overwrite objects.
        5. Clean up objects and buckets.
   */

    public static S3ActionsWrapper _s3ActionsWrapper = null!;
    public static IConfiguration _configuration = null!;
    private static string _resourcePrefix = null!;
    private static string noLockBucketName = null!;
    private static string lockEnabledBucketName = null!;
    private static string retentionAfterCreationBucketName = null!;
    private static List<string> bucketNames = new List<string>();
    private static List<string> fileNames = new List<string>();

    public static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonS3>()
                    .AddTransient<S3ActionsWrapper>()
```

```
            )
            .Build();

        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();

        ConfigurationSetup();

        ServicesSetup(host);

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Amazon Simple Storage Service (S3)
Object Locking Workflow Scenario.");
            Console.WriteLine(new string('-', 80));
            await Setup(true);

            await DemoActionChoices();

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Cleaning up resources.");
            Console.WriteLine(new string('-', 80));
            await Cleanup(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Amazon S3 Object Locking Workflow is complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem: {ex.Message}");
            await Cleanup(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Populate the services for use within the console application.
```

```
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _s3ActionsWrapper = host.Services.GetRequiredService<S3ActionsWrapper>();
    }

    /// <summary>
    /// Any setup operations needed.
    /// </summary>
    public static void ConfigurationSetup()
    {
        _resourcePrefix = _configuration["resourcePrefix"] ?? "dotnet-example";

        noLockBucketName = _resourcePrefix + "-no-lock";
        lockEnabledBucketName = _resourcePrefix + "-lock-enabled";
        retentionAfterCreationBucketName = _resourcePrefix + "-retention-after-
creation";

        bucketNames.Add(noLockBucketName);
        bucketNames.Add(lockEnabledBucketName);
        bucketNames.Add(retentionAfterCreationBucketName);
    }

    // <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Setup(bool interactive)
    {
        Console.WriteLine(
            "\nFor this workflow, we will use the AWS SDK for .NET to create
 several S3\n" +
            "buckets and files to demonstrate working with S3 locking features.
\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you are ready to start.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nS3 buckets can be created either with or without
 object lock enabled.");
```

```
        await _s3ActionsWrapper.CreateBucketWithObjectLock(noLockBucketName,
false);
        await _s3ActionsWrapper.CreateBucketWithObjectLock(lockEnabledBucketName,
true);
        await
_s3ActionsWrapper.CreateBucketWithObjectLock(retentionAfterCreationBucketName,
false);

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nA bucket can be configured to use object locking
with a default retention period.");
        await
_s3ActionsWrapper.ModifyBucketDefaultRetention(retentionAfterCreationBucketName,
true,
            ObjectLockRetentionMode.Governance, DateTime.UtcNow.AddDays(1));

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nObject lock policies can also be added to existing
buckets.");
        await _s3ActionsWrapper.EnableObjectLockOnBucket(lockEnabledBucketName);

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        // Upload some files to the buckets.
        Console.WriteLine("\nNow let's add some test files:");
        var fileName = _configuration["exampleFileName"] ?? "exampleFile.txt";
        int fileCount = 2;
        // Create the file if it does not already exist.
        if (!File.Exists(fileName))
        {
            await using StreamWriter sw = File.CreateText(fileName);
            await sw.WriteLineAsync(
                "This is a sample file for uploading to a bucket.");
        }

        foreach (var bucketName in bucketNames)
```

```
        {
            for (int i = 0; i < fileCount; i++)
            {
                var numberedFileName = Path.GetFileNameWithoutExtension(fileName)
+ i + Path.GetExtension(fileName);
                fileNames.Add(numberedFileName);
                await _s3ActionsWrapper.UploadFileAsync(bucketName,
numberedFileName, fileName);
            }
        }
        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        if (!interactive)
            return true;
        Console.WriteLine("\nNow we can set some object lock policies on
individual files:");
        foreach (var bucketName in bucketNames)
        {
            for (int i = 0; i < fileNames.Count; i++)
            {
                // No modifications to the objects in the first bucket.
                if (bucketName != bucketNames[0])
                {
                    var exampleFileName = fileNames[i];
                    switch (i)
                    {
                        case 0:
                            {
                                var question =
                                    $"\nWould you like to add a legal hold to
{exampleFileName} in {bucketName}? (y/n)";
                                if (GetYesNoResponse(question))
                                {
                                    // Set a legal hold.
                                    await
_s3ActionsWrapper.ModifyObjectLegalHold(bucketName, exampleFileName,
ObjectLockLegalHoldStatus.On);

                                }
                                break;
                            }
                        case 1:
```

```
                            {
                                var question =
                                    $"\nWould you like to add a 1 day Governance
retention period to {exampleFileName} in {bucketName}? (y/n)" +
                                    "\nReminder: Only a user with the
s3:BypassGovernanceRetention permission will be able to delete this file or its
bucket until the retention period has expired.";
                                if (GetYesNoResponse(question))
                                {
                                    // Set a Governance mode retention period for
1 day.
                                    await
_s3ActionsWrapper.ModifyObjectRetentionPeriod(
                                        bucketName, exampleFileName,
                                        ObjectLockRetentionMode.Governance,
                                        DateTime.UtcNow.AddDays(1));
                                }
                                break;
                            }
                        }
                    }
                }
            }
            Console.WriteLine(new string('-', 80));
            return true;
        }

        // <summary>
        /// List all of the current buckets and objects.
        /// </summary>
        /// <param name="interactive">True to run as interactive.</param>
        /// <returns>The list of buckets and objects.</returns>
        public static async Task<List<S3ObjectVersion>> ListBucketsAndObjects(bool
interactive)
        {
            var allObjects = new List<S3ObjectVersion>();
            foreach (var bucketName in bucketNames)
            {
                var objectsInBucket = await
_s3ActionsWrapper.ListBucketObjectsAndVersions(bucketName);
                foreach (var objectKey in objectsInBucket.Versions)
                {
                    allObjects.Add(objectKey);
                }
```

```
        }

        if (interactive)
        {
            Console.WriteLine("\nCurrent buckets and objects:\n");
            int i = 0;
            foreach (var bucketObject in allObjects)
            {
                i++;
                Console.WriteLine(
                    $"{i}: {bucketObject.Key} \n\tBucket:
{bucketObject.BucketName}\n\tVersion: {bucketObject.VersionId}");
            }
        }

        return allObjects;
    }

    /// <summary>
    /// Present the user with the demo action choices.
    /// </summary>
    /// <returns>Async task.</returns>
    public static async Task<bool> DemoActionChoices()
    {
        var choices = new string[]{
            "List all files in buckets.",
            "Attempt to delete a file.",
            "Attempt to delete a file with retention period bypass.",
            "Attempt to overwrite a file.",
            "View the object and bucket retention settings for a file.",
            "View the legal hold settings for a file.",
            "Finish the workflow."};

        var choice = 0;
        // Keep asking the user until they choose to move on.
        while (choice != 6)
        {
            Console.WriteLine(new string('-', 80));
            choice = GetChoiceResponse(
                "\nExplore the S3 locking features by selecting one of the
following choices:"
                , choices);
            Console.WriteLine(new string('-', 80));
            switch (choice)
```

```
            {
                case 0:
                    {
                        await ListBucketsAndObjects(true);
                        break;
                    }
                case 1:
                    {
                        Console.WriteLine("\nEnter the number of the object to
delete:");
                        var allFiles = await ListBucketsAndObjects(true);
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.DeleteObjectFromBucket(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, false, allFiles[fileChoice].VersionId);
                        break;
                    }
                case 2:
                    {
                        Console.WriteLine("\nEnter the number of the object to
delete:");
                        var allFiles = await ListBucketsAndObjects(true);
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.DeleteObjectFromBucket(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, true, allFiles[fileChoice].VersionId);
                        break;
                    }
                case 3:
                    {
                        var allFiles = await ListBucketsAndObjects(true);
                        Console.WriteLine("\nEnter the number of the object to
overwrite:");
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        // Create the file if it does not already exist.
                        if (!File.Exists(allFiles[fileChoice].Key))
                        {
                            await using StreamWriter sw =
File.CreateText(allFiles[fileChoice].Key);
                            await sw.WriteLineAsync(
```

```
                                "This is a sample file for uploading to a
bucket.");
                        }
                        await
_s3ActionsWrapper.UploadFileAsync(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, allFiles[fileChoice].Key);
                        break;
                    }
                case 4:
                    {
                        var allFiles = await ListBucketsAndObjects(true);
                        Console.WriteLine("\nEnter the number of the object and
bucket to view:");
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.GetObjectRetention(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key);
                        await
_s3ActionsWrapper.GetBucketObjectLockConfiguration(allFiles[fileChoice].BucketName);
                        break;
                    }
                case 5:
                    {
                        var allFiles = await ListBucketsAndObjects(true);
                        Console.WriteLine("\nEnter the number of the object to
view:");
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.GetObjectLegalHold(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key);
                        break;
                    }
            }
        }
        return true;
    }

    // <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
```

```
   public static async Task<bool> Cleanup(bool interactive)
   {
       Console.WriteLine(new string('-', 80));

       if (!interactive || GetYesNoResponse("Do you want to clean up all files
and buckets? (y/n) "))
       {
           // Remove all locks and delete all buckets and objects.
           var allFiles = await ListBucketsAndObjects(false);
           foreach (var fileInfo in allFiles)
           {
               // Check for a legal hold.
               var legalHold = await
_s3ActionsWrapper.GetObjectLegalHold(fileInfo.BucketName, fileInfo.Key);
               if (legalHold?.Status?.Value == ObjectLockLegalHoldStatus.On)
               {
                   await
_s3ActionsWrapper.ModifyObjectLegalHold(fileInfo.BucketName, fileInfo.Key,
ObjectLockLegalHoldStatus.Off);
               }

               // Check for a retention period.
               var retention = await
_s3ActionsWrapper.GetObjectRetention(fileInfo.BucketName, fileInfo.Key);
               var hasRetentionPeriod = retention?.Mode ==
ObjectLockRetentionMode.Governance && retention.RetainUntilDate >
DateTime.UtcNow.Date;
               await
_s3ActionsWrapper.DeleteObjectFromBucket(fileInfo.BucketName, fileInfo.Key,
hasRetentionPeriod, fileInfo.VersionId);
           }

           foreach (var bucketName in bucketNames)
           {
               await _s3ActionsWrapper.DeleteBucketByName(bucketName);
           }

       }
       else
       {
           Console.WriteLine(
               "Ok, we'll leave the resources intact.\n" +
               "Don't forget to delete them when you're done with them or you
might incur unexpected charges."
```

```
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Helper method to get a yes or no response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <returns>True if the user responds with a yes.</returns>
    private static bool GetYesNoResponse(string question)
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null && ynResponse.Equals("y",
 StringComparison.InvariantCultureIgnoreCase);
        return response;
    }

    /// <summary>
    /// Helper method to get a choice response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <param name="choices">The choices to print on the console.</param>
    /// <returns>The index of the selected choice</returns>
    private static int GetChoiceResponse(string? question, string[] choices)
    {
        if (question != null)
        {
            Console.WriteLine(question);

            for (int i = 0; i < choices.Length; i++)
            {
                Console.WriteLine($"\t{i + 1}. {choices[i]}");
            }
        }

        var choiceNumber = 0;
        while (choiceNumber < 1 || choiceNumber > choices.Length)
        {
```

```
            var choice = Console.ReadLine();
            Int32.TryParse(choice, out choiceNumber);
        }

        return choiceNumber - 1;
    }
}
```

Eine Wrapper-Klasse für S3-Funktionen.

```
using System.Net;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

namespace S3ObjectLockScenario;

/// <summary>
/// Encapsulate the Amazon S3 operations.
/// </summary>
public class S3ActionsWrapper
{
    private readonly IAmazonS3 _amazonS3;

    /// <summary>
    /// Constructor for the S3ActionsWrapper.
    /// </summary>
    /// <param name="amazonS3">The injected S3 client.</param>
    public S3ActionsWrapper(IAmazonS3 amazonS3, IConfiguration configuration)
    {
        _amazonS3 = amazonS3;
    }

    /// <summary>
    /// Create a new Amazon S3 bucket with object lock actions.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to create.</param>
    /// <param name="enableObjectLock">True to enable object lock on the
 bucket.</param>
    /// <returns>True if successful.</returns>
```

```
   public async Task<bool> CreateBucketWithObjectLock(string bucketName, bool
enableObjectLock)
   {
       Console.WriteLine($"\tCreating bucket {bucketName} with object lock
{enableObjectLock}.");
       try
       {
           var request = new PutBucketRequest
           {
               BucketName = bucketName,
               UseClientRegion = true,
               ObjectLockEnabledForBucket = enableObjectLock,
           };

           var response = await _amazonS3.PutBucketAsync(request);

           return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
       }
       catch (AmazonS3Exception ex)
       {
           Console.WriteLine($"Error creating bucket: '{ex.Message}'");
           return false;
       }
   }

   /// <summary>
   /// Enable object lock on an existing bucket.
   /// </summary>
   /// <param name="bucketName">The name of the bucket to modify.</param>
   /// <returns>True if successful.</returns>
   public async Task<bool> EnableObjectLockOnBucket(string bucketName)
   {
       try
       {
           // First, enable Versioning on the bucket.
           await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
           {
               BucketName = bucketName,
               VersioningConfig = new S3BucketVersioningConfig()
               {
                   EnableMfaDelete = false,
                   Status = VersionStatus.Enabled
               }
```

```
            });

            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled("Enabled"),
                },
            };

            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded an object lock policy to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error modifying object lock: '{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Set or modify a retention period on an object in an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The key of the object.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date retention expires.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyObjectRetentionPeriod(string bucketName,
        string objectKey, ObjectLockRetentionMode retention, DateTime
retainUntilDate)
    {
        try
        {
            var request = new PutObjectRetentionRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                Retention = new ObjectLockRetention()
                {
```

```
                    Mode = retention,
                    RetainUntilDate = retainUntilDate
                }
            };

            var response = await _amazonS3.PutObjectRetentionAsync(request);
            Console.WriteLine($"\tSet retention for {objectKey} in {bucketName}
until {retainUntilDate:d}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying retention period:
'{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Set or modify a retention period on an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket to modify.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date for retention until.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyBucketDefaultRetention(string bucketName, bool
enableObjectLock, ObjectLockRetentionMode retention, DateTime retainUntilDate)
    {
        var enabledString = enableObjectLock ? "Enabled" : "Disabled";
        var timeDifference = retainUntilDate.Subtract(DateTime.Now);
        try
        {
            // First, enable Versioning on the bucket.
            await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
            {
                BucketName = bucketName,
                VersioningConfig = new S3BucketVersioningConfig()
                {
                    EnableMfaDelete = false,
                    Status = VersionStatus.Enabled
                }
            });
```

```
            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled(enabledString),
                    Rule = new ObjectLockRule()
                    {
                        DefaultRetention = new DefaultRetention()
                        {
                            Mode = retention,
                            Days = timeDifference.Days // Can be specified in
days or years but not both.
                        }
                    }
                }
            };

            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded a default retention to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying object lock: '{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Get the retention period for an S3 object.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object retention details.</returns>
    public async Task<ObjectLockRetention> GetObjectRetention(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectRetentionRequest()
            {
```

```
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectRetentionAsync(request);
            Console.WriteLine($"\tObject retention for {objectKey} in
{bucketName}: " +
                              $"\n\t{response.Retention.Mode} until
{response.Retention.RetainUntilDate:d}.");
            return response.Retention;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch object lock retention:
'{ex.Message}'");
            return new ObjectLockRetention();
        }
    }

    /// <summary>
    /// Set or modify a legal hold on an object in an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The key of the object.</param>
    /// <param name="holdStatus">The On or Off status for the legal hold.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyObjectLegalHold(string bucketName,
        string objectKey, ObjectLockLegalHoldStatus holdStatus)
    {
        try
        {
            var request = new PutObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                LegalHold = new ObjectLockLegalHold()
                {
                    Status = holdStatus
                }
            };

            var response = await _amazonS3.PutObjectLegalHoldAsync(request);
            Console.WriteLine($"\tModified legal hold for {objectKey} in
{bucketName}.");
```

```
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying legal hold: '{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Get the legal hold details for an S3 object.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object legal hold details.</returns>
    public async Task<ObjectLockLegalHold> GetObjectLegalHold(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectLegalHoldAsync(request);
            Console.WriteLine($"\tObject legal hold for {objectKey} in
{bucketName}: " +
                              $"\n\tStatus: {response.LegalHold.Status}");
            return response.LegalHold;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch legal hold: '{ex.Message}'");
            return new ObjectLockLegalHold();
        }
    }

    /// <summary>
    /// Get the object lock configuration details for an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket to get details.</param>
    /// <returns>The bucket's object lock configuration details.</returns>
```

```
   public async Task<ObjectLockConfiguration>
GetBucketObjectLockConfiguration(string bucketName)
   {
       try
       {
           var request = new GetObjectLockConfigurationRequest()
           {
               BucketName = bucketName
           };

           var response = await
_amazonS3.GetObjectLockConfigurationAsync(request);
           Console.WriteLine($"\tBucket object lock config for {bucketName} in
{bucketName}: " +
                             $"\n\tEnabled:
{response.ObjectLockConfiguration.ObjectLockEnabled}" +
                             $"\n\tRule:
{response.ObjectLockConfiguration.Rule?.DefaultRetention}");

           return response.ObjectLockConfiguration;
       }
       catch (AmazonS3Exception ex)
       {
           Console.WriteLine($"\tUnable to fetch object lock config:
'{ex.Message}'");
           return new ObjectLockConfiguration();
       }
   }

   /// <summary>
   /// Upload a file from the local computer to an Amazon S3 bucket.
   /// </summary>
   /// <param name="bucketName">The Amazon S3 bucket to use.</param>
   /// <param name="objectName">The object to upload.</param>
   /// <param name="filePath">The path, including file name, of the object to
upload.</param>
   /// <returns>True if success.<returns>
   public async Task<bool> UploadFileAsync(string bucketName, string objectName,
string filePath)
   {
       var request = new PutObjectRequest
       {
           BucketName = bucketName,
           Key = objectName,
```

```
            FilePath = filePath,
            ChecksumAlgorithm = ChecksumAlgorithm.SHA256
        };

        var response = await _amazonS3.PutObjectAsync(request);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"\tSuccessfully uploaded {objectName} to
 {bucketName}.");
            return true;
        }
        else
        {
            Console.WriteLine($"\tCould not upload {objectName} to
 {bucketName}.");
            return false;
        }
    }

    /// <summary>
    /// List bucket objects and versions.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <returns>The list of objects and versions.</returns>
    public async Task<ListVersionsResponse> ListBucketObjectsAndVersions(string
 bucketName)
    {
        var request = new ListVersionsRequest()
        {
            BucketName = bucketName
        };

        var response = await _amazonS3.ListVersionsAsync(request);
        return response;
    }

    /// <summary>
    /// Delete an object from a specific bucket.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <param name="objectKey">The key of the object to delete.</param>
    /// <param name="hasRetention">True if the object has retention settings.</
param>
    /// <param name="versionId">Optional versionId.</param>
```

```
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteObjectFromBucket(string bucketName, string
objectKey, bool hasRetention, string? versionId = null)
    {
        try
        {
            var request = new DeleteObjectRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                VersionId = versionId,
            };
            if (hasRetention)
            {
                // Set the BypassGovernanceRetention header
                // if the file has retention settings.
                request.BypassGovernanceRetention = true;
            }
            await _amazonS3.DeleteObjectAsync(request);
            Console.WriteLine(
                $"Deleted {objectKey} in {bucketName}.");
            return true;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to delete object {objectKey} in bucket
{bucketName}: " + ex.Message);
            return false;
        }
    }

    /// <summary>
    /// Delete a specific bucket.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <param name="objectKey">The key of the object to delete.</param>
    /// <param name="versionId">Optional versionId.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteBucketByName(string bucketName)
    {
        try
        {
            var request = new DeleteBucketRequest() { BucketName = bucketName, };
            var response = await _amazonS3.DeleteBucketAsync(request);
```

```
            Console.WriteLine($"\tDelete for {bucketName} complete.");
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to delete bucket {bucketName}: " +
 ex.Message);
            return false;
        }

    }

}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - GetObjectLegalHold
  - GetObjectLockConfiguration
  - GetObjectRetention
  - PutObjectLegalHold
  - PutObjectLockConfiguration
  - PutObjectRetention

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwalten von Zugriffskontrolllisten (ACLs) für Amazon S3-Buckets mithilfe eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie eine neue Zugriffssteuerungsliste (ACL) für Amazon-S3-Buckets verwalten.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to manage Amazon Simple Storage Service
/// (Amazon S3) access control lists (ACLs) to control Amazon S3 bucket
/// access.
/// </summary>
public class ManageACLs
{
    public static async Task Main()
    {
        string bucketName = "doc-example-bucket1";
        string newBucketName = "doc-example-bucket2";
        string keyName = "sample-object.txt";
        string emailAddress = "someone@example.com";

        // If the AWS Region where your bucket is located is different from
        // the Region defined for the default user, pass the Amazon S3
bucket's
        // name to the client constructor. It should look like this:
        // RegionEndpoint bucketRegion = RegionEndpoint.USEast1;
        IAmazonS3 client = new AmazonS3Client();

        await TestBucketObjectACLsAsync(client, bucketName, newBucketName,
keyName, emailAddress);
    }

    /// <summary>
```

```
        /// Creates a new Amazon S3 bucket with a canned ACL, then retrieves the
ACL
        /// information and then adds a new ACL to one of the objects in the
        /// Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// methods to create a bucket, get an ACL, and add a different ACL to
        /// one of the objects.</param>
        /// <param name="bucketName">A string representing the original Amazon S3
        /// bucket name.</param>
        /// <param name="newBucketName">A string representing the name of the
        /// new bucket that will be created.</param>
        /// <param name="keyName">A string representing the key name of an Amazon
S3
        /// object for which we will change the ACL.</param>
        /// <param name="emailAddress">A string representing the email address
        /// belonging to the person to whom access to the Amazon S3 bucket will
be
        /// granted.</param>
        public static async Task TestBucketObjectACLsAsync(
            IAmazonS3 client,
            string bucketName,
            string newBucketName,
            string keyName,
            string emailAddress)
        {
            try
            {
                // Create a new Amazon S3 bucket and specify canned ACL.
                var success = await CreateBucketWithCannedACLAsync(client,
newBucketName);

                // Get the ACL on a bucket.
                await GetBucketACLAsync(client, bucketName);

                // Add (replace) the ACL on an object in a bucket.
                await AddACLToExistingObjectAsync(client, bucketName, keyName,
emailAddress);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine($"Exception: {amazonS3Exception.Message}");
            }
```

```
        }

        /// <summary>
        /// Creates a new Amazon S3 bucket with a canned ACL attached.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// PutBucketAsync.</param>
        /// <param name="newBucketName">A string representing the name of the
        /// new Amazon S3 bucket.</param>
        /// <returns>Returns a boolean value indicating success or failure.</
returns>
        public static async Task<bool> CreateBucketWithCannedACLAsync(IAmazonS3
 client, string newBucketName)
        {
            var request = new PutBucketRequest()
            {
                BucketName = newBucketName,
                BucketRegion = S3Region.EUWest1,

                // Add a canned ACL.
                CannedACL = S3CannedACL.LogDeliveryWrite,
            };

            var response = await client.PutBucketAsync(request);
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }


        /// <summary>
        /// Retrieves the ACL associated with the Amazon S3 bucket name in the
        /// bucketName parameter.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// PutBucketAsync.</param>
        /// <param name="bucketName">The Amazon S3 bucket for which we want to
 get the
        /// ACL list.</param>
        /// <returns>Returns an S3AccessControlList returned from the call to
        /// GetACLAsync.</returns>
        public static async Task<S3AccessControlList> GetBucketACLAsync(IAmazonS3
 client, string bucketName)
        {
            GetACLResponse response = await client.GetACLAsync(new GetACLRequest
            {
```

```
                BucketName = bucketName,
            });

            return response.AccessControlList;
        }



        /// <summary>
        /// Adds a new ACL to an existing object in the Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// PutBucketAsync.</param>
        /// <param name="bucketName">A string representing the name of the Amazon
 S3
        /// bucket containing the object to which we want to apply a new ACL.</
param>
        /// <param name="keyName">A string representing the name of the object
        /// to which we want to apply the new ACL.</param>
        /// <param name="emailAddress">The email address of the person to whom
        /// we will be applying to whom access will be granted.</param>
        public static async Task AddACLToExistingObjectAsync(IAmazonS3 client,
 string bucketName, string keyName, string emailAddress)
        {
            // Retrieve the ACL for an object.
            GetACLResponse aclResponse = await client.GetACLAsync(new
 GetACLRequest
            {
                BucketName = bucketName,
                Key = keyName,
            });

            S3AccessControlList acl = aclResponse.AccessControlList;

            // Retrieve the owner.
            Owner owner = acl.Owner;

            // Clear existing grants.
            acl.Grants.Clear();

            // Add a grant to reset the owner's full permission
            // (the previous clear statement removed all permissions).
            var fullControlGrant = new S3Grant
            {
```

```
                Grantee = new S3Grantee { CanonicalUser = acl.Owner.Id },
            };
            acl.AddGrant(fullControlGrant.Grantee, S3Permission.FULL_CONTROL);

            // Specify email to identify grantee for granting permissions.
            var grantUsingEmail = new S3Grant
            {
                Grantee = new S3Grantee { EmailAddress = emailAddress },
                Permission = S3Permission.WRITE_ACP,
            };

            // Specify log delivery group as grantee.
            var grantLogDeliveryGroup = new S3Grant
            {
                Grantee = new S3Grantee { URI = "http://acs.amazonaws.com/groups/
s3/LogDelivery" },
                Permission = S3Permission.WRITE,
            };

            // Create a new ACL.
            var newAcl = new S3AccessControlList
            {
                Grants = new List<S3Grant> { grantUsingEmail,
 grantLogDeliveryGroup },
                Owner = owner,
            };

            // Set the new ACL. We're throwing away the response here.
            _ = await client.PutACLAsync(new PutACLRequest
            {
                BucketName = bucketName,
                Key = keyName,
                AccessControlList = newAcl,
            });
        }

    }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - GetBucketAcl
  - GetObjectAcl

- PutBucketAcl
- PutObjectAcl

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Versionierte Amazon S3-Objekte in Batches mit einer Lambda-Funktion mithilfe eines AWS -SDK verwalten

Das folgende Codebeispiel zeigt, wie versionierte S3-Objekte in Batches mit einer Lambda-Funktion verwaltet werden.

Python

SDK für Python (Boto3)

Zeigt, wie versionierte Objekte von Amazon Simple Storage Service (Amazon S3) in Batches manipuliert werden, indem Aufträge erstellt werden, die - AWS Lambda Funktionen zur Ausführung der Verarbeitung aufrufen. In diesem Beispiel wird ein versionsfähiger Bucket erstellt, der die Strophen des Gedichts You Are Old, Father William von Lewis Carroll hochlädt und Amazon-S3-Batch-Aufträge verwendet, um das Gedicht auf verschiedene Arten zu verdrehen.

Lernen Sie Folgendes:

- Erstellen Sie Lambda-Funktionen, die mit versionierten Objekten arbeiten.
- Erstellen Sie ein Manifest von zu aktualisierenden Objekten.
- Erstellen Sie Batch-Aufträge, die Lambda-Funktionen zum Aktualisieren von Objekten aufrufen.
- Löschen Sie Lambda-Funktionen.
- Leeren und löschen Sie einen versionierten Bucket.

Dieses Beispiel wird am besten auf angezeigt GitHub. Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon S3

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Analysieren von Amazon S3-URIs mit einem AWS -SDK

Das folgende Codebeispiel zeigt, wie Sie Amazon S3-URIs analysieren, um wichtige Komponenten wie den Bucket-Namen und Objektschlüssel zu extrahieren.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Analysieren Sie eine Amazon S3-URI mithilfe der S3Uri-Klasse.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.S3Uri;
import software.amazon.awssdk.services.s3.S3Utilities;

import java.net.URI;
import java.util.List;
import java.util.Map;

    /**
     *
     * @param s3Client    - An S3Client through which you acquire an S3Uri
 instance.
     * @param s3ObjectUrl - A complex URL (String) that is used to demonstrate
 S3Uri
     *                    capabilities.
     */
    public static void parseS3UriExample(S3Client s3Client, String s3ObjectUrl) {
        logger.info(s3ObjectUrl);
```

```
        // Console output:
        // 'https://s3.us-west-1.amazonaws.com/myBucket/resources/doc.txt?
versionId=abc123&partNumber=77&partNumber=88'.

        // Create an S3Utilities object using the configuration of the s3Client.
        S3Utilities s3Utilities = s3Client.utilities();

        // From a String URL create a URI object to pass to the parseUri()
 method.
        URI uri = URI.create(s3ObjectUrl);
        S3Uri s3Uri = s3Utilities.parseUri(uri);

        // If the URI contains no value for the Region, bucket or key, the SDK
 returns
        // an empty Optional.
        // The SDK returns decoded URI values.

        Region region = s3Uri.region().orElse(null);
        log("region", region);
        // Console output: 'region: us-west-1'.

        String bucket = s3Uri.bucket().orElse(null);
        log("bucket", bucket);
        // Console output: 'bucket: myBucket'.

        String key = s3Uri.key().orElse(null);
        log("key", key);
        // Console output: 'key: resources/doc.txt'.

        Boolean isPathStyle = s3Uri.isPathStyle();
        log("isPathStyle", isPathStyle);
        // Console output: 'isPathStyle: true'.

        // If the URI contains no query parameters, the SDK returns an empty map.
        Map<String, List<String>> queryParams = s3Uri.rawQueryParameters();
        log("rawQueryParameters", queryParams);
        // Console output: 'rawQueryParameters: {versionId=[abc123],
 partNumber=[77,
        // 88]}'.

        // Retrieve the first or all values for a query parameter as shown in the
        // following code.
        String versionId =
 s3Uri.firstMatchingRawQueryParameter("versionId").orElse(null);
```

```
        log("firstMatchingRawQueryParameter-versionId", versionId);
        // Console output: 'firstMatchingRawQueryParameter-versionId: abc123'.

        String partNumber =
 s3Uri.firstMatchingRawQueryParameter("partNumber").orElse(null);
        log("firstMatchingRawQueryParameter-partNumber", partNumber);
        // Console output: 'firstMatchingRawQueryParameter-partNumber: 77'.

        List<String> partNumbers =
 s3Uri.firstMatchingRawQueryParameters("partNumber");
        log("firstMatchingRawQueryParameter", partNumbers);
        // Console output: 'firstMatchingRawQueryParameter: [77, 88]'.

        /*
         * Object keys and query parameters with reserved or unsafe characters,
 must be
         * URL-encoded.
         * For example replace whitespace " " with "%20".
         * Valid:
         * "https://s3.us-west-1.amazonaws.com/myBucket/object%20key?query=
%5Bbrackets%5D"
         * Invalid:
         * "https://s3.us-west-1.amazonaws.com/myBucket/object key?
query=[brackets]"
         *
         * Virtual-hosted-style URIs with bucket names that contain a dot, ".",
 the dot
         * must not be URL-encoded.
         * Valid: "https://my.Bucket.s3.us-west-1.amazonaws.com/key"
         * Invalid: "https://my%2EBucket.s3.us-west-1.amazonaws.com/key"
         */
    }

    private static void log(String s3UriElement, Object element) {
        if (element == null) {
            logger.info("{}: {}", s3UriElement, "null");
        } else {
            logger.info("{}: {}", s3UriElement, element.toString());
        }
    }
```

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Ausführen einer mehrteiligen Kopie eines Amazon S3-Objekts mithilfe eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie eine mehrteilige Kopie eines Amazon-S3-Objekts erstellen.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to perform a multi-part copy from one Amazon
/// Simple Storage Service (Amazon S3) bucket to another.
/// </summary>
public class MPUapiCopyObj
{
    private const string SourceBucket = "doc-example-bucket1";
    private const string TargetBucket = "doc-example-bucket2";
    private const string SourceObjectKey = "example.mov";
    private const string TargetObjectKey = "copied_video_file.mov";

    /// <summary>
    /// This method starts the multi-part upload.
    /// </summary>
    public static async Task Main()
    {
```

```
            var s3Client = new AmazonS3Client();
            Console.WriteLine("Copying object...");
            await MPUCopyObjectAsync(s3Client);
        }

        /// <summary>
        /// This method uses the passed client object to perform a multipart
        /// copy operation.
        /// </summary>
        /// <param name="client">An Amazon S3 client object that will be used
        /// to perform the copy.</param>
        public static async Task MPUCopyObjectAsync(AmazonS3Client client)
        {
            // Create a list to store the copy part responses.
            var copyResponses = new List<CopyPartResponse>();

            // Setup information required to initiate the multipart upload.
            var initiateRequest = new InitiateMultipartUploadRequest
            {
                BucketName = TargetBucket,
                Key = TargetObjectKey,
            };

            // Initiate the upload.
            InitiateMultipartUploadResponse initResponse =
                await client.InitiateMultipartUploadAsync(initiateRequest);

            // Save the upload ID.
            string uploadId = initResponse.UploadId;

            try
            {
                // Get the size of the object.
                var metadataRequest = new GetObjectMetadataRequest
                {
                    BucketName = SourceBucket,
                    Key = SourceObjectKey,
                };

                GetObjectMetadataResponse metadataResponse =
                    await client.GetObjectMetadataAsync(metadataRequest);
                var objectSize = metadataResponse.ContentLength; // Length in
bytes.
```

```
                // Copy the parts.
                var partSize = 5 * (long)Math.Pow(2, 20); // Part size is 5 MB.

                long bytePosition = 0;
                for (int i = 1; bytePosition < objectSize; i++)
                {
                    var copyRequest = new CopyPartRequest
                    {
                        DestinationBucket = TargetBucket,
                        DestinationKey = TargetObjectKey,
                        SourceBucket = SourceBucket,
                        SourceKey = SourceObjectKey,
                        UploadId = uploadId,
                        FirstByte = bytePosition,
                        LastByte = bytePosition + partSize - 1 >= objectSize ?
objectSize - 1 : bytePosition + partSize - 1,
                        PartNumber = i,
                    };

                    copyResponses.Add(await client.CopyPartAsync(copyRequest));

                    bytePosition += partSize;
                }

                // Set up to complete the copy.
                var completeRequest = new CompleteMultipartUploadRequest
                {
                    BucketName = TargetBucket,
                    Key = TargetObjectKey,
                    UploadId = initResponse.UploadId,
                };
                completeRequest.AddPartETags(copyResponses);

                // Complete the copy.
                CompleteMultipartUploadResponse completeUploadResponse =
                    await client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error encountered on server.
Message:'{e.Message}' when writing an object");
            }
            catch (Exception e)
            {
```

```
                Console.WriteLine($"Unknown encountered on server.
Message:'{e.Message}' when writing an object");
            }
        }
    }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - CompleteMultipartUpload
  - CreateMultipartUpload
  - GetObjectMetadata
  - UploadPartCopy

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Durchführen eines mehrteiligen Uploads in ein Amazon S3-Objekt mithilfe eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie einen mehrteiligen Upload in ein Amazon-S3-Objekt durchführen.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In den Codebeispielen werden folgende Importe verwendet.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
```

```
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;

import java.io.IOException;
import java.io.RandomAccessFile;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.ByteBuffer;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
import java.util.UUID;
```

Verwenden Sie den S3-Transfer-Manager zusätzlich zum AWS -CRT-basierten S3-Client, um einen mehrteiligen Upload auf transparente Weise durchzuführen, wenn die Größe des Inhalts einen Schwellenwert überschreitet. Der Standardschwellenwert beträgt 8 MB.

```
    public void multipartUploadWithTransferManager(String filePath) {
        S3TransferManager transferManager = S3TransferManager.create();
        UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
                .putObjectRequest(b -> b
                        .bucket(bucketName)
                        .key(key))
                .source(Paths.get(filePath))
                .build();
        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);
        fileUpload.completionFuture().join();
        transferManager.close();
    }
```

Verwenden Sie die S3Client-API oder (S3AsyncClient API), um einen mehrteiligen Upload durchzuführen.

```
    public void multipartUploadWithS3Client(String filePath) {

        // Initiate the multipart upload.
        CreateMultipartUploadResponse createMultipartUploadResponse =
s3Client.createMultipartUpload(b -> b
                .bucket(bucketName)
                .key(key));
        String uploadId = createMultipartUploadResponse.uploadId();

        // Upload the parts of the file.
        int partNumber = 1;
        List<CompletedPart> completedParts = new ArrayList<>();
        ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

        try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
            long fileSize = file.length();
            int position = 0;
            while (position < fileSize) {
                file.seek(position);
                int read = file.getChannel().read(bb);

                bb.flip(); // Swap position and limit before reading from the
buffer.
                UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                        .bucket(bucketName)
                        .key(key)
                        .uploadId(uploadId)
                        .partNumber(partNumber)
                        .build();

                UploadPartResponse partResponse = s3Client.uploadPart(
                        uploadPartRequest,
                        RequestBody.fromByteBuffer(bb));

                CompletedPart part = CompletedPart.builder()
                        .partNumber(partNumber)
                        .eTag(partResponse.eTag())
                        .build();
                completedParts.add(part);
```

```
                bb.clear();
                position += read;
                partNumber++;
            }
        } catch (IOException e) {
            logger.error(e.getMessage());
        }

        // Complete the multipart upload.
        s3Client.completeMultipartUpload(b -> b
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)

.multipartUpload(CompletedMultipartUpload.builder().parts(completedParts).build()));
    }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - CompleteMultipartUpload
  - CreateMultipartUpload
  - UploadPart

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Hochladen oder Herunterladen großer Dateien in und von Amazon S3 mithilfe eines - AWS SDK

Die folgenden Codebeispiele veranschaulichen, wie Sie große Dateien zu S3 hochladen bzw. von S3 herunterladen.

Weitere Informationen finden Sie unter Hochladen eines Objekts mit Multipart-Upload.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie Funktionen auf, die Dateien mithilfe von Amazon S3 TransferUtility3 in und aus einem S3-Bucket übertragen.

```
global using System.Text;
global using Amazon.S3;
global using Amazon.S3.Model;
global using Amazon.S3.Transfer;
global using TransferUtilityBasics;



// This Amazon S3 client uses the default user credentials
// defined for this computer.
using Microsoft.Extensions.Configuration;

IAmazonS3 client = new AmazonS3Client();
var transferUtil = new TransferUtility(client);
IConfiguration _configuration;

_configuration = new ConfigurationBuilder()
    .SetBasePath(Directory.GetCurrentDirectory())
    .AddJsonFile("settings.json") // Load test settings from JSON file.
    .AddJsonFile("settings.local.json",
        true) // Optionally load local settings.
    .Build();

// Edit the values in settings.json to use an S3 bucket and files that
// exist on your AWS account and on the local computer where you
// run this scenario.
var bucketName = _configuration["BucketName"];
```

```
var localPath =
 $"{Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData)}\
\TransferFolder";

DisplayInstructions();

PressEnter();

Console.WriteLine();

// Upload a single file to an S3 bucket.
DisplayTitle("Upload a single file");

var fileToUpload = _configuration["FileToUpload"];
Console.WriteLine($"Uploading {fileToUpload} to the S3 bucket, {bucketName}.");

var success = await TransferMethods.UploadSingleFileAsync(transferUtil,
 bucketName, fileToUpload, localPath);
if (success)
{
    Console.WriteLine($"Successfully uploaded the file, {fileToUpload} to
 {bucketName}.");
}

PressEnter();

// Upload a local directory to an S3 bucket.
DisplayTitle("Upload all files from a local directory");
Console.WriteLine("Upload all the files in a local folder to an S3 bucket.");
const string keyPrefix = "UploadFolder";
var uploadPath = $"{localPath}\\UploadFolder";

Console.WriteLine($"Uploading the files in {uploadPath} to {bucketName}");
DisplayTitle($"{uploadPath} files");
DisplayLocalFiles(uploadPath);
Console.WriteLine();

PressEnter();

success = await TransferMethods.UploadFullDirectoryAsync(transferUtil,
 bucketName, keyPrefix, uploadPath);
if (success)
{
```

```
    Console.WriteLine($"Successfully uploaded the files in {uploadPath} to
 {bucketName}.");
    Console.WriteLine($"{bucketName} currently contains the following files:");
    await DisplayBucketFiles(client, bucketName, keyPrefix);
    Console.WriteLine();
}

PressEnter();

// Download a single file from an S3 bucket.
DisplayTitle("Download a single file");
Console.WriteLine("Now we will download a single file from an S3 bucket.");

var keyName = _configuration["FileToDownload"];

Console.WriteLine($"Downloading {keyName} from {bucketName}.");

success = await TransferMethods.DownloadSingleFileAsync(transferUtil, bucketName,
 keyName, localPath);
if (success)
{
    Console.WriteLine("$Successfully downloaded the file, {keyName} from
 {bucketName}.");
}

PressEnter();

// Download the contents of a directory from an S3 bucket.
DisplayTitle("Download the contents of an S3 bucket");
var s3Path = _configuration["S3Path"];
var downloadPath = $"{localPath}\\{s3Path}";

Console.WriteLine($"Downloading the contents of {bucketName}\\{s3Path}");
Console.WriteLine($"{bucketName}\\{s3Path} contains the following files:");
await DisplayBucketFiles(client, bucketName, s3Path);
Console.WriteLine();

success = await TransferMethods.DownloadS3DirectoryAsync(transferUtil,
 bucketName, s3Path, downloadPath);
if (success)
{
    Console.WriteLine($"Downloaded the files in {bucketName} to
 {downloadPath}.");
    Console.WriteLine($"{downloadPath} now contains the following files:");
```

```
    DisplayLocalFiles(downloadPath);
}

Console.WriteLine("\nThe TransferUtility Basics application has completed.");
PressEnter();

// Displays the title for a section of the scenario.
static void DisplayTitle(string titleText)
{
    var sepBar = new string('-', Console.WindowWidth);

    Console.WriteLine(sepBar);
    Console.WriteLine(CenterText(titleText));
    Console.WriteLine(sepBar);
}

// Displays a description of the actions to be performed by the scenario.
static void DisplayInstructions()
{
    var sepBar = new string('-', Console.WindowWidth);

    DisplayTitle("Amazon S3 Transfer Utility Basics");
    Console.WriteLine("This program shows how to use the Amazon S3 Transfer
 Utility.");
    Console.WriteLine("It performs the following actions:");
    Console.WriteLine("\t1. Upload a single object to an S3 bucket.");
    Console.WriteLine("\t2. Upload an entire directory from the local computer to
 an\n\t  S3 bucket.");
    Console.WriteLine("\t3. Download a single object from an S3 bucket.");
    Console.WriteLine("\t4. Download the objects in an S3 bucket to a local
 directory.");
    Console.WriteLine($"\n{sepBar}");
}

// Pauses the scenario.
static void PressEnter()
{
    Console.WriteLine("Press <Enter> to continue.");
    _ = Console.ReadLine();
    Console.WriteLine("\n");
}

// Returns the string textToCenter, padded on the left with spaces
// that center the text on the console display.
```

```
static string CenterText(string textToCenter)
{
    var centeredText = new StringBuilder();
    var screenWidth = Console.WindowWidth;
    centeredText.Append(new string(' ', (int)(screenWidth -
 textToCenter.Length) / 2));
    centeredText.Append(textToCenter);
    return centeredText.ToString();
}

// Displays a list of file names included in the specified path.
static void DisplayLocalFiles(string localPath)
{
    var fileList = Directory.GetFiles(localPath);
    if (fileList.Length > 0)
    {
        foreach (var fileName in fileList)
        {
            Console.WriteLine(fileName);
        }
    }
}

// Displays a list of the files in the specified S3 bucket and prefix.
static async Task DisplayBucketFiles(IAmazonS3 client, string bucketName, string
 s3Path)
{
    ListObjectsV2Request request = new()
    {
        BucketName = bucketName,
        Prefix = s3Path,
        MaxKeys = 5,
    };

    var response = new ListObjectsV2Response();

    do
    {
        response = await client.ListObjectsV2Async(request);

        response.S3Objects
            .ForEach(obj => Console.WriteLine($"{obj.Key}"));

        // If the response is truncated, set the request ContinuationToken
```

```
        // from the NextContinuationToken property of the response.
        request.ContinuationToken = response.NextContinuationToken;
    } while (response.IsTruncated);
}
```

Laden Sie eine einzelne Datei hoch.

```
        /// <summary>
        /// Uploads a single file from the local computer to an S3 bucket.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The name of the S3 bucket where the file
        /// will be stored.</param>
        /// <param name="fileName">The name of the file to upload.</param>
        /// <param name="localPath">The local path where the file is stored.</
param>
        /// <returns>A boolean value indicating the success of the action.</
returns>
        public static async Task<bool> UploadSingleFileAsync(
            TransferUtility transferUtil,
            string bucketName,
            string fileName,
            string localPath)
        {
            if (File.Exists($"{localPath}\\{fileName}"))
            {
                try
                {
                    await transferUtil.UploadAsync(new
 TransferUtilityUploadRequest
                    {
                        BucketName = bucketName,
                        Key = fileName,
                        FilePath = $"{localPath}\\{fileName}",
                    });

                    return true;
                }
                catch (AmazonS3Exception s3Ex)
```

```
                {
                    Console.WriteLine($"Could not upload {fileName} from
 {localPath} because:");
                    Console.WriteLine(s3Ex.Message);
                    return false;
                }
            }
            else
            {
                Console.WriteLine($"{fileName} does not exist in {localPath}");
                return false;
            }
        }
```

Laden Sie ein ganzes lokales Verzeichnis hoch.

```
        /// <summary>
        /// Uploads all the files in a local directory to a directory in an S3
        /// bucket.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The name of the S3 bucket where the files
        /// will be stored.</param>
        /// <param name="keyPrefix">The key prefix is the S3 directory where
        /// the files will be stored.</param>
        /// <param name="localPath">The local directory that contains the files
        /// to be uploaded.</param>
        /// <returns>A Boolean value representing the success of the action.</
returns>
        public static async Task<bool> UploadFullDirectoryAsync(
            TransferUtility transferUtil,
            string bucketName,
            string keyPrefix,
            string localPath)
        {
            if (Directory.Exists(localPath))
            {
                try
                {
```

```
                    await transferUtil.UploadDirectoryAsync(new
 TransferUtilityUploadDirectoryRequest
                    {
                        BucketName = bucketName,
                        KeyPrefix = keyPrefix,
                        Directory = localPath,
                    });

                    return true;
                }
                catch (AmazonS3Exception s3Ex)
                {
                    Console.WriteLine($"Can't upload the contents of {localPath}
 because:");
                    Console.WriteLine(s3Ex?.Message);
                    return false;
                }
            }
            else
            {
                Console.WriteLine($"The directory {localPath} does not exist.");
                return false;
            }
        }
```

Laden Sie eine einzelne Datei herunter.

```
        /// <summary>
        /// Download a single file from an S3 bucket to the local computer.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The name of the S3 bucket containing the
        /// file to download.</param>
        /// <param name="keyName">The name of the file to download.</param>
        /// <param name="localPath">The path on the local computer where the
        /// downloaded file will be saved.</param>
        /// <returns>A Boolean value indicating the results of the action.</
returns>
        public static async Task<bool> DownloadSingleFileAsync(
```

```
        TransferUtility transferUtil,
            string bucketName,
            string keyName,
            string localPath)
        {
            await transferUtil.DownloadAsync(new TransferUtilityDownloadRequest
            {
                BucketName = bucketName,
                Key = keyName,
                FilePath = $"{localPath}\\{keyName}",
            });

            return (File.Exists($"{localPath}\\{keyName}"));
        }
```

Laden Sie den Inhalt eines S3 Buckets herunter.

```
        /// <summary>
        /// Downloads the contents of a directory in an S3 bucket to a
        /// directory on the local computer.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The bucket containing the files to
 download.</param>
        /// <param name="s3Path">The S3 directory where the files are located.</
param>
        /// <param name="localPath">The local path to which the files will be
        /// saved.</param>
        /// <returns>A Boolean value representing the success of the action.</
returns>
        public static async Task<bool> DownloadS3DirectoryAsync(
            TransferUtility transferUtil,
            string bucketName,
            string s3Path,
            string localPath)
        {
            int fileCount = 0;

            // If the directory doesn't exist, it will be created.
```

```
            if (Directory.Exists(s3Path))
            {
                var files = Directory.GetFiles(localPath);
                fileCount = files.Length;
            }

            await transferUtil.DownloadDirectoryAsync(new
TransferUtilityDownloadDirectoryRequest
            {
                BucketName = bucketName,
                LocalDirectory = localPath,
                S3Directory = s3Path,
            });

            if (Directory.Exists(localPath))
            {
                var files = Directory.GetFiles(localPath);
                if (files.Length > fileCount)
                {
                    return true;
                }

                // No change in the number of files. Assume
                // the download failed.
                return false;
            }

            // The local directory doesn't exist. No files
            // were downloaded.
            return false;
        }
```

Verfolgen Sie den Fortschritt eines Uploads mithilfe der TransferUtility.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Transfer;

    /// <summary>
    /// This example shows how to track the progress of a multipart upload
```

```
    /// using the Amazon Simple Storage Service (Amazon S3) TransferUtility to
    /// upload to an Amazon S3 bucket.
    /// </summary>
    public class TrackMPUUsingHighLevelAPI
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "sample_pic.png";
            string path = "filepath/directory/";
            string filePath = $"{path}{keyName}";

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the Amazon S3 client object's constructor.
            // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
            IAmazonS3 client = new AmazonS3Client();

            await TrackMPUAsync(client, bucketName, filePath, keyName);
        }

        /// <summary>
        /// Starts an Amazon S3 multipart upload and assigns an event handler to
        /// track the progress of the upload.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
        /// perform the multipart upload.</param>
        /// <param name="bucketName">The name of the bucket to which to upload
        /// the file.</param>
        /// <param name="filePath">The path, including the file name of the
        /// file to be uploaded to the Amazon S3 bucket.</param>
        /// <param name="keyName">The file name to be used in the
        /// destination Amazon S3 bucket.</param>
        public static async Task TrackMPUAsync(
            IAmazonS3 client,
            string bucketName,
            string filePath,
            string keyName)
        {
            try
            {
                var fileTransferUtility = new TransferUtility(client);

                // Use TransferUtilityUploadRequest to configure options.
```

```
                // In this example we subscribe to an event.
                var uploadRequest =
                    new TransferUtilityUploadRequest
                    {
                        BucketName = bucketName,
                        FilePath = filePath,
                        Key = keyName,
                    };

                uploadRequest.UploadProgressEvent +=
                    new EventHandler<UploadProgressArgs>(
                        UploadRequest_UploadPartProgressEvent);

                await fileTransferUtility.UploadAsync(uploadRequest);
                Console.WriteLine("Upload completed");
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error:: {ex.Message}");
            }
        }

        /// <summary>
        /// Event handler to check the progress of the multipart upload.
        /// </summary>
        /// <param name="sender">The object that raised the event.</param>
        /// <param name="e">The object that contains multipart upload
        /// information.</param>
        public static void UploadRequest_UploadPartProgressEvent(object sender,
UploadProgressArgs e)
        {
            // Process event.
            Console.WriteLine($"{e.TransferredBytes}/{e.TotalBytes}");
        }
    }
```

Laden Sie ein Objekt mit Verschlüsselung hoch.

```
    using System;
    using System.Collections.Generic;
    using System.IO;
```

```
using System.Security.Cryptography;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// Uses the Amazon Simple Storage Service (Amazon S3) low level API to
/// perform a multipart upload to an Amazon S3 bucket.
/// </summary>
public class SSECLowLevelMPUcopyObject
{
    public static async Task Main()
    {
        string existingBucketName = "doc-example-bucket";
        string sourceKeyName = "sample_file.txt";
        string targetKeyName = "sample_file_copy.txt";
        string filePath = $"sample\\{targetKeyName}";

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USEast1.
        IAmazonS3 client = new AmazonS3Client();

        // Create the encryption key.
        var base64Key = CreateEncryptionKey();

        await CreateSampleObjUsingClientEncryptionKeyAsync(
            client,
            existingBucketName,
            sourceKeyName,
            filePath,
            base64Key);
    }

    /// <summary>
    /// Creates the encryption key to use with the multipart upload.
    /// </summary>
    /// <returns>A string containing the base64-encoded key for encrypting
    /// the multipart upload.</returns>
    public static string CreateEncryptionKey()
    {
        Aes aesEncryption = Aes.Create();
        aesEncryption.KeySize = 256;
```

```
            aesEncryption.GenerateKey();
            string base64Key = Convert.ToBase64String(aesEncryption.Key);
            return base64Key;
        }

        /// <summary>
        /// Creates and uploads an object using a multipart upload.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 object used to
        /// initialize and perform the multipart upload.</param>
        /// <param name="existingBucketName">The name of the bucket to which
        /// the object will be uploaded.</param>
        /// <param name="sourceKeyName">The source object name.</param>
        /// <param name="filePath">The location of the source object.</param>
        /// <param name="base64Key">The encryption key to use with the upload.</
param>
        public static async Task CreateSampleObjUsingClientEncryptionKeyAsync(
            IAmazonS3 client,
            string existingBucketName,
            string sourceKeyName,
            string filePath,
            string base64Key)
        {
            List<UploadPartResponse> uploadResponses = new
 List<UploadPartResponse>();

            InitiateMultipartUploadRequest initiateRequest = new
 InitiateMultipartUploadRequest
            {
                BucketName = existingBucketName,
                Key = sourceKeyName,
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            InitiateMultipartUploadResponse initResponse =
               await client.InitiateMultipartUploadAsync(initiateRequest);

            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
```

```
                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                    {
                        BucketName = existingBucketName,
                        Key = sourceKeyName,
                        UploadId = initResponse.UploadId,
                        PartNumber = i,
                        PartSize = partSize,
                        FilePosition = filePosition,
                        FilePath = filePath,
                        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key,
                    };

                    // Upload part and add response to our list.
                    uploadResponses.Add(await
client.UploadPartAsync(uploadRequest));

                    filePosition += partSize;
                }

                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId,
                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (Exception exception)
            {
                Console.WriteLine($"Exception occurred: {exception.Message}");

                // If there was an error, abort the multipart upload.
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
```

```
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId,
                };

                await client.AbortMultipartUploadAsync(abortMPURequest);
            }
        }
    }
```

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Laden Sie ein großes Objekt hoch, indem Sie einen Upload-Manager verwenden, um die Daten in Teile zu zerlegen und sie gleichzeitig hochzuladen.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// UploadLargeObject uses an upload manager to upload data to an object in a
 bucket.
```

```
// The upload manager breaks large data into parts and uploads the parts
 concurrently.
func (basics BucketBasics) UploadLargeObject(bucketName string, objectKey string,
 largeObject []byte) error {
 largeBuffer := bytes.NewReader(largeObject)
 var partMiBs int64 = 10
 uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
  u.PartSize = partMiBs * 1024 * 1024
 })
 _, err := uploader.Upload(context.TODO(), &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
  Body:   largeBuffer,
 })
 if err != nil {
  log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }

 return err
}
```

Laden Sie ein großes Objekt herunter, indem Sie einen Download-Manager verwenden, um die Daten in Teilen abzurufen und sie gleichzeitig herunterzuladen.

```
// DownloadLargeObject uses a download manager to download an object from a
 bucket.
// The download manager gets the data in parts and writes them to a buffer until
 all of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(bucketName string, objectKey
 string) ([]byte, error) {
 var partMiBs int64 = 10
 downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader)
 {
  d.PartSize = partMiBs * 1024 * 1024
 })
 buffer := manager.NewWriteAtBuffer([]byte{})
 _, err := downloader.Download(context.TODO(), buffer, &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
```

```
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return buffer.Bytes(), err
}
```

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie Funktionen auf, die Dateien mithilfe von S3 in und aus einem S3S3TransferManager-Bucket übertragen.

```
    public Integer downloadObjectsToDirectory(S3TransferManager transferManager,
            URI destinationPathURI, String bucketName) {
        DirectoryDownload directoryDownload =
transferManager.downloadDirectory(DownloadDirectoryRequest.builder()
                .destination(Paths.get(destinationPathURI))
                .bucket(bucketName)
                .build());
        CompletedDirectoryDownload completedDirectoryDownload =
directoryDownload.completionFuture().join();

        completedDirectoryDownload.failedTransfers()
                .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
        return completedDirectoryDownload.failedTransfers().size();
    }
```

Laden Sie ein ganzes lokales Verzeichnis hoch.

```
   public Integer uploadDirectory(S3TransferManager transferManager,
           URI sourceDirectory, String bucketName) {
       DirectoryUpload directoryUpload =
transferManager.uploadDirectory(UploadDirectoryRequest.builder()
               .source(Paths.get(sourceDirectory))
               .bucket(bucketName)
               .build());

       CompletedDirectoryUpload completedDirectoryUpload =
directoryUpload.completionFuture().join();
       completedDirectoryUpload.failedTransfers()
               .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
       return completedDirectoryUpload.failedTransfers().size();
   }
```

Laden Sie eine einzelne Datei hoch.

```
   public String uploadFile(S3TransferManager transferManager, String
bucketName,
                            String key, URI filePathURI) {
       UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
           .putObjectRequest(b -> b.bucket(bucketName).key(key))
           .addTransferListener(LoggingTransferListener.create())
           .source(Paths.get(filePathURI))
           .build();

       FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);

       CompletedFileUpload uploadResult = fileUpload.completionFuture().join();
       return uploadResult.response().eTag();
   }
```

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Laden Sie eine große Datei hoch.

```
import {
  CreateMultipartUploadCommand,
  UploadPartCommand,
  CompleteMultipartUploadCommand,
  AbortMultipartUploadCommand,
  S3Client,
} from "@aws-sdk/client-s3";

const twentyFiveMB = 25 * 1024 * 1024;

export const createString = (size = twentyFiveMB) => {
  return "x".repeat(size);
};

export const main = async () => {
  const s3Client = new S3Client({});
  const bucketName = "test-bucket";
  const key = "multipart.txt";
  const str = createString();
  const buffer = Buffer.from(str, "utf8");

  let uploadId;

  try {
    const multipartUpload = await s3Client.send(
      new CreateMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
      }),
    );
```

```
    uploadId = multipartUpload.UploadId;

    const uploadPromises = [];
    // Multipart uploads require a minimum size of 5 MB per part.
    const partSize = Math.ceil(buffer.length / 5);

    // Upload each part.
    for (let i = 0; i < 5; i++) {
      const start = i * partSize;
      const end = start + partSize;
      uploadPromises.push(
        s3Client
          .send(
            new UploadPartCommand({
              Bucket: bucketName,
              Key: key,
              UploadId: uploadId,
              Body: buffer.subarray(start, end),
              PartNumber: i + 1,
            }),
          )
          .then((d) => {
            console.log("Part", i + 1, "uploaded");
            return d;
          }),
      );
    }

    const uploadResults = await Promise.all(uploadPromises);

    return await s3Client.send(
      new CompleteMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
        UploadId: uploadId,
        MultipartUpload: {
          Parts: uploadResults.map(({ ETag }, i) => ({
            ETag,
            PartNumber: i + 1,
          })),
        },
      }),
    );
```

```
    // Verify the output by downloading the file from the Amazon Simple Storage
 Service (Amazon S3) console.
    // Because the output is a 25 MB string, text editors might struggle to open
 the file.
  } catch (err) {
    console.error(err);

    if (uploadId) {
      const abortCommand = new AbortMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
        UploadId: uploadId,
      });

      await s3Client.send(abortCommand);
    }
  }
};
```

Laden Sie eine große Datei herunter.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { createWriteStream } from "fs";

const s3Client = new S3Client({});
const oneMB = 1024 * 1024;

export const getObjectRange = ({ bucket, key, start, end }) => {
  const command = new GetObjectCommand({
    Bucket: bucket,
    Key: key,
    Range: `bytes=${start}-${end}`,
  });

  return s3Client.send(command);
};

export const getRangeAndLength = (contentRange) => {
  const [range, length] = contentRange.split("/");
  const [start, end] = range.split("-");
  return {
    start: parseInt(start),
```

```
    end: parseInt(end),
    length: parseInt(length),
  };
};

export const isComplete = ({ end, length }) => end === length - 1;

// When downloading a large file, you might want to break it down into
// smaller pieces. Amazon S3 accepts a Range header to specify the start
// and end of the byte range to be downloaded.
const downloadInChunks = async ({ bucket, key }) => {
  const writeStream = createWriteStream(
    fileURLToPath(new URL(`./${key}`, import.meta.url))
  ).on("error", (err) => console.error(err));

  let rangeAndLength = { start: -1, end: -1, length: -1 };

  while (!isComplete(rangeAndLength)) {
    const { end } = rangeAndLength;
    const nextRange = { start: end + 1, end: end + oneMB };

    console.log(`Downloading bytes ${nextRange.start} to ${nextRange.end}`);

    const { ContentRange, Body } = await getObjectRange({
      bucket,
      key,
      ...nextRange,
    });

    writeStream.write(await Body.transformToByteArray());
    rangeAndLength = getRangeAndLength(ContentRange);
  }
};

export const main = async () => {
  await downloadInChunks({
    bucket: "my-cool-bucket",
    key: "my-cool-object.txt",
  });
};
```

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die Dateien mit mehreren der verfügbaren Transfer-Manager-Einstellungen übertragen. Verwenden Sie eine Callback-Klasse, um während der Dateiübertragung Callback-Fortschritte zu schreiben.

```
import sys
import threading

import boto3
from boto3.s3.transfer import TransferConfig


MB = 1024 * 1024
s3 = boto3.resource("s3")


class TransferCallback:
    """
    Handle callbacks from the transfer manager.

    The transfer manager periodically calls the __call__ method throughout
    the upload and download process so that it can take action, such as
    displaying progress to the user and collecting data about the transfer.
    """

    def __init__(self, target_size):
        self._target_size = target_size
        self._total_transferred = 0
        self._lock = threading.Lock()
        self.thread_info = {}

    def __call__(self, bytes_transferred):
        """
```

```
        The callback method that is called by the transfer manager.

        Display progress during file transfer and collect per-thread transfer
        data. This method can be called by multiple threads, so shared instance
        data is protected by a thread lock.
        """
        thread = threading.current_thread()
        with self._lock:
            self._total_transferred += bytes_transferred
            if thread.ident not in self.thread_info.keys():
                self.thread_info[thread.ident] = bytes_transferred
            else:
                self.thread_info[thread.ident] += bytes_transferred

            target = self._target_size * MB
            sys.stdout.write(
                f"\r{self._total_transferred} of {target} transferred "
                f"({(self._total_transferred / target) * 100:.2f}%)."
            )
            sys.stdout.flush()


def upload_with_default_configuration(
    local_file_path, bucket_name, object_key, file_size_mb
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, using the default
    configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def upload_with_chunksize_and_meta(
    local_file_path, bucket_name, object_key, file_size_mb, metadata=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart chunk size and adding metadata to the Amazon S3 object.

    The multipart chunk size controls the size of the chunks of data that are
```

```
    sent in the request. A smaller chunk size typically results in the transfer
    manager using more threads for the upload.

    The metadata is a set of key-value pairs that are stored with the object
    in Amazon S3.
    """
    transfer_callback = TransferCallback(file_size_mb)

    config = TransferConfig(multipart_chunksize=1 * MB)
    extra_args = {"Metadata": metadata} if metadata else None
    s3.Bucket(bucket_name).upload_file(
        local_file_path,
        object_key,
        Config=config,
        ExtraArgs=extra_args,
        Callback=transfer_callback,
    )
    return transfer_callback.thread_info


def upload_with_high_threshold(local_file_path, bucket_name, object_key,
 file_size_mb):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard upload instead of
    a multipart upload.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def upload_with_sse(
    local_file_path, bucket_name, object_key, file_size_mb, sse_key=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, adding server-side
    encryption with customer-provided encryption keys to the object.
```

```
    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)
    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey":
 sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, ExtraArgs=extra_args,
 Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_default_configuration(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using the
    default configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_single_thread(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using a
    single thread.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(use_threads=False)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
```

```
    return transfer_callback.thread_info


def download_with_high_threshold(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard download instead
    of a multipart download.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_sse(
    bucket_name, object_key, download_file_path, file_size_mb, sse_key
):
    """
    Download a file from an Amazon S3 bucket to a local folder, adding a
    customer-provided encryption key to the request.

    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)

    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey":
 sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, ExtraArgs=extra_args, Callback=transfer_callback
    )
    return transfer_callback.thread_info
```

Demonstrieren Sie die Funktionen des Transfer-Managers und melden Sie Ergebnisse.

```
import hashlib
import os
import platform
import shutil
import time

import boto3
from boto3.s3.transfer import TransferConfig
from botocore.exceptions import ClientError
from botocore.exceptions import ParamValidationError
from botocore.exceptions import NoCredentialsError

import file_transfer

MB = 1024 * 1024
# These configuration attributes affect both uploads and downloads.
CONFIG_ATTRS = (
    "multipart_threshold",
    "multipart_chunksize",
    "max_concurrency",
    "use_threads",
)
# These configuration attributes affect only downloads.
DOWNLOAD_CONFIG_ATTRS = ("max_io_queue", "io_chunksize", "num_download_attempts")


class TransferDemoManager:
    """
    Manages the demonstration. Collects user input from a command line, reports
    transfer results, maintains a list of artifacts created during the
    demonstration, and cleans them up after the demonstration is completed.
    """

    def __init__(self):
        self._s3 = boto3.resource("s3")
        self._chore_list = []
        self._create_file_cmd = None
```

```
        self._size_multiplier = 0
        self.file_size_mb = 30
        self.demo_folder = None
        self.demo_bucket = None
        self._setup_platform_specific()
        self._terminal_width = shutil.get_terminal_size(fallback=(80, 80))[0]

    def collect_user_info(self):
        """
        Collect local folder and Amazon S3 bucket name from the user. These
        locations are used to store files during the demonstration.
        """
        while not self.demo_folder:
            self.demo_folder = input(
                "Which file folder do you want to use to store " "demonstration
files? "
            )
            if not os.path.isdir(self.demo_folder):
                print(f"{self.demo_folder} isn't a folder!")
                self.demo_folder = None

        while not self.demo_bucket:
            self.demo_bucket = input(
                "Which Amazon S3 bucket do you want to use to store "
                "demonstration files? "
            )
            try:
                self._s3.meta.client.head_bucket(Bucket=self.demo_bucket)
            except ParamValidationError as err:
                print(err)
                self.demo_bucket = None
            except ClientError as err:
                print(err)
                print(
                    f"Either {self.demo_bucket} doesn't exist or you don't "
                    f"have access to it."
                )
                self.demo_bucket = None

    def demo(
        self, question, upload_func, download_func, upload_args=None,
download_args=None
    ):
        """Run a demonstration.
```

```
        Ask the user if they want to run this specific demonstration.
        If they say yes, create a file on the local path, upload it
        using the specified upload function, then download it using the
        specified download function.
        """
        if download_args is None:
            download_args = {}
        if upload_args is None:
            upload_args = {}
        question = question.format(self.file_size_mb)
        answer = input(f"{question} (y/n)")
        if answer.lower() == "y":
            local_file_path, object_key, download_file_path =
self._create_demo_file()

            file_transfer.TransferConfig = self._config_wrapper(
                TransferConfig, CONFIG_ATTRS
            )
            self._report_transfer_params(
                "Uploading", local_file_path, object_key, **upload_args
            )
            start_time = time.perf_counter()
            thread_info = upload_func(
                local_file_path,
                self.demo_bucket,
                object_key,
                self.file_size_mb,
                **upload_args,
            )
            end_time = time.perf_counter()
            self._report_transfer_result(thread_info, end_time - start_time)

            file_transfer.TransferConfig = self._config_wrapper(
                TransferConfig, CONFIG_ATTRS + DOWNLOAD_CONFIG_ATTRS
            )
            self._report_transfer_params(
                "Downloading", object_key, download_file_path, **download_args
            )
            start_time = time.perf_counter()
            thread_info = download_func(
                self.demo_bucket,
                object_key,
                download_file_path,
```

```
                self.file_size_mb,
                **download_args,
            )
            end_time = time.perf_counter()
            self._report_transfer_result(thread_info, end_time - start_time)

    def last_name_set(self):
        """Get the name set used for the last demo."""
        return self._chore_list[-1]

    def cleanup(self):
        """
        Remove files from the demo folder, and uploaded objects from the
        Amazon S3 bucket.
        """
        print("-" * self._terminal_width)
        for local_file_path, s3_object_key, downloaded_file_path in
self._chore_list:
            print(f"Removing {local_file_path}")
            try:
                os.remove(local_file_path)
            except FileNotFoundError as err:
                print(err)

            print(f"Removing {downloaded_file_path}")
            try:
                os.remove(downloaded_file_path)
            except FileNotFoundError as err:
                print(err)

            if self.demo_bucket:
                print(f"Removing {self.demo_bucket}:{s3_object_key}")
                try:

self._s3.Bucket(self.demo_bucket).Object(s3_object_key).delete()
                except ClientError as err:
                    print(err)

    def _setup_platform_specific(self):
        """Set up platform-specific command used to create a large file."""
        if platform.system() == "Windows":
            self._create_file_cmd = "fsutil file createnew {} {}"
            self._size_multiplier = MB
        elif platform.system() == "Linux" or platform.system() == "Darwin":
```

```
            self._create_file_cmd = f"dd if=/dev/urandom of={{}} " f"bs={MB}
count={{}}"
            self._size_multiplier = 1
        else:
            raise EnvironmentError(
                f"Demo of platform {platform.system()} isn't supported."
            )

    def _create_demo_file(self):
        """
        Create a file in the demo folder specified by the user. Store the local
        path, object name, and download path for later cleanup.

        Only the local file is created by this method. The Amazon S3 object and
        download file are created later during the demonstration.

        Returns:
        A tuple that contains the local file path, object name, and download
        file path.
        """
        file_name_template = "TestFile{}-{}.demo"
        local_suffix = "local"
        object_suffix = "s3object"
        download_suffix = "downloaded"
        file_tag = len(self._chore_list) + 1

        local_file_path = os.path.join(
            self.demo_folder, file_name_template.format(file_tag, local_suffix)
        )

        s3_object_key = file_name_template.format(file_tag, object_suffix)

        downloaded_file_path = os.path.join(
            self.demo_folder, file_name_template.format(file_tag,
download_suffix)
        )

        filled_cmd = self._create_file_cmd.format(
            local_file_path, self.file_size_mb * self._size_multiplier
        )

        print(
            f"Creating file of size {self.file_size_mb} MB "
            f"in {self.demo_folder} by running:"
```

```
        )
        print(f"{'':4}{filled_cmd}")
        os.system(filled_cmd)

        chore = (local_file_path, s3_object_key, downloaded_file_path)
        self._chore_list.append(chore)
        return chore

    def _report_transfer_params(self, verb, source_name, dest_name, **kwargs):
        """Report configuration and extra arguments used for a file transfer."""
        print("-" * self._terminal_width)
        print(f"{verb} {source_name} ({self.file_size_mb} MB) to {dest_name}")
        if kwargs:
            print("With extra args:")
            for arg, value in kwargs.items():
                print(f'{"":4}{arg:<20}: {value}')

    @staticmethod
    def ask_user(question):
        """
        Ask the user a yes or no question.

        Returns:
        True when the user answers 'y' or 'Y'; otherwise, False.
        """
        answer = input(f"{question} (y/n) ")
        return answer.lower() == "y"

    @staticmethod
    def _config_wrapper(func, config_attrs):
        def wrapper(*args, **kwargs):
            config = func(*args, **kwargs)
            print("With configuration:")
            for attr in config_attrs:
                print(f'{"":4}{attr:<20}: {getattr(config, attr)}')
            return config

        return wrapper

    @staticmethod
    def _report_transfer_result(thread_info, elapsed):
        """Report the result of a transfer, including per-thread data."""
        print(f"\nUsed {len(thread_info)} threads.")
        for ident, byte_count in thread_info.items():
```

```
            print(f"{'':4}Thread {ident} copied {byte_count} bytes.")
        print(f"Your transfer took {elapsed:.2f} seconds.")


def main():
    """
    Run the demonstration script for s3_file_transfer.
    """
    demo_manager = TransferDemoManager()
    demo_manager.collect_user_info()

    # Upload and download with default configuration. Because the file is 30 MB
    # and the default multipart_threshold is 8 MB, both upload and download are
    # multipart transfers.
    demo_manager.demo(
        "Do you want to upload and download a {} MB file "
        "using the default configuration?",
        file_transfer.upload_with_default_configuration,
        file_transfer.download_with_default_configuration,
    )

    # Upload and download with multipart_threshold set higher than the size of
    # the file. This causes the transfer manager to use standard transfers
    # instead of multipart transfers.
    demo_manager.demo(
        "Do you want to upload and download a {} MB file "
        "as a standard (not multipart) transfer?",
        file_transfer.upload_with_high_threshold,
        file_transfer.download_with_high_threshold,
    )

    # Upload with specific chunk size and additional metadata.
    # Download with a single thread.
    demo_manager.demo(
        "Do you want to upload a {} MB file with a smaller chunk size and "
        "then download the same file using a single thread?",
        file_transfer.upload_with_chunksize_and_meta,
        file_transfer.download_with_single_thread,
        upload_args={
            "metadata": {
                "upload_type": "chunky",
                "favorite_color": "aqua",
                "size": "medium",
            }
```

```
        },
    )

    # Upload using server-side encryption with customer-provided
    # encryption keys.
    # Generate a 256-bit key from a passphrase.
    sse_key = hashlib.sha256("demo_passphrase".encode("utf-8")).digest()
    demo_manager.demo(
        "Do you want to upload and download a {} MB file using "
        "server-side encryption?",
        file_transfer.upload_with_sse,
        file_transfer.download_with_sse,
        upload_args={"sse_key": sse_key},
        download_args={"sse_key": sse_key},
    )

    # Download without specifying an encryption key to show that the
    # encryption key must be included to download an encrypted object.
    if demo_manager.ask_user(
        "Do you want to try to download the encrypted "
        "object without sending the required key?"
    ):
        try:
            _, object_key, download_file_path = demo_manager.last_name_set()
            file_transfer.download_with_default_configuration(
                demo_manager.demo_bucket,
                object_key,
                download_file_path,
                demo_manager.file_size_mb,
            )
        except ClientError as err:
            print(
                "Got expected error when trying to download an encrypted "
                "object without specifying encryption info:"
            )
            print(f"{'':4}{err}")

    # Remove all created and downloaded files, remove all objects from
    # S3 storage.
    if demo_manager.ask_user(
        "Demonstration complete. Do you want to remove local files " "and S3
objects?"
    ):
        demo_manager.cleanup()
```

```
if __name__ == "__main__":
    try:
        main()
    except NoCredentialsError as error:
        print(error)
        print(
            "To run this example, you must have valid credentials in "
            "a shared credential file or set in environment variables."
        )
```

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
use std::fs::File;
use std::io::prelude::*;
use std::path::Path;

use aws_config::meta::region::RegionProviderChain;
use aws_sdk_s3::error::DisplayErrorContext;
use aws_sdk_s3::operation::{
    create_multipart_upload::CreateMultipartUploadOutput,
 get_object::GetObjectOutput,
};
use aws_sdk_s3::types::{CompletedMultipartUpload, CompletedPart};
use aws_sdk_s3::{config::Region, Client as S3Client};
use aws_smithy_types::byte_stream::{ByteStream, Length};
use rand::distributions::Alphanumeric;
use rand::{thread_rng, Rng};
use s3_service::error::Error;
use std::process;
```

```
use uuid::Uuid;

//In bytes, minimum chunk size of 5MB. Increase CHUNK_SIZE to send larger chunks.
const CHUNK_SIZE: u64 = 1024 * 1024 * 5;
const MAX_CHUNKS: u64 = 10000;

#[tokio::main]
pub async fn main() {
    if let Err(err) = run_example().await {
        eprintln!("Error: {}", DisplayErrorContext(err));
        process::exit(1);
    }
}

async fn run_example() -> Result<(), Error> {
    let shared_config = aws_config::load_from_env().await;
    let client = S3Client::new(&shared_config);

    let bucket_name = format!("doc-example-bucket-{}", Uuid::new_v4());
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));
    let region = region_provider.region().await.unwrap();
    s3_service::create_bucket(&client, &bucket_name, region.as_ref()).await?;

    let key = "sample.txt".to_string();
    let multipart_upload_res: CreateMultipartUploadOutput = client
        .create_multipart_upload()
        .bucket(&bucket_name)
        .key(&key)
        .send()
        .await
        .unwrap();
    let upload_id = multipart_upload_res.upload_id().unwrap();

    //Create a file of random characters for the upload.
    let mut file = File::create(&key).expect("Could not create sample file.");
    // Loop until the file is 5 chunks.
    while file.metadata().unwrap().len() <= CHUNK_SIZE * 4 {
        let rand_string: String = thread_rng()
            .sample_iter(&Alphanumeric)
            .take(256)
            .map(char::from)
            .collect();
        let return_string: String = "\n".to_string();
```

```
        file.write_all(rand_string.as_ref())
            .expect("Error writing to file.");
        file.write_all(return_string.as_ref())
            .expect("Error writing to file.");
    }

    let path = Path::new(&key);
    let file_size = tokio::fs::metadata(path)
        .await
        .expect("it exists I swear")
        .len();

    let mut chunk_count = (file_size / CHUNK_SIZE) + 1;
    let mut size_of_last_chunk = file_size % CHUNK_SIZE;
    if size_of_last_chunk == 0 {
        size_of_last_chunk = CHUNK_SIZE;
        chunk_count -= 1;
    }

    if file_size == 0 {
        panic!("Bad file size.");
    }
    if chunk_count > MAX_CHUNKS {
        panic!("Too many chunks! Try increasing your chunk size.")
    }

    let mut upload_parts: Vec<CompletedPart> = Vec::new();

    for chunk_index in 0..chunk_count {
        let this_chunk = if chunk_count - 1 == chunk_index {
            size_of_last_chunk
        } else {
            CHUNK_SIZE
        };
        let stream = ByteStream::read_from()
            .path(path)
            .offset(chunk_index * CHUNK_SIZE)
            .length(Length::Exact(this_chunk))
            .build()
            .await
            .unwrap();
        //Chunk index needs to start at 0, but part numbers start at 1.
        let part_number = (chunk_index as i32) + 1;
        let upload_part_res = client
```

```
            .upload_part()
            .key(&key)
            .bucket(&bucket_name)
            .upload_id(upload_id)
            .body(stream)
            .part_number(part_number)
            .send()
            .await?;
        upload_parts.push(
            CompletedPart::builder()
                .e_tag(upload_part_res.e_tag.unwrap_or_default())
                .part_number(part_number)
                .build(),
        );
    }
    let completed_multipart_upload: CompletedMultipartUpload =
CompletedMultipartUpload::builder()
        .set_parts(Some(upload_parts))
        .build();

    let _complete_multipart_upload_res = client
        .complete_multipart_upload()
        .bucket(&bucket_name)
        .key(&key)
        .multipart_upload(completed_multipart_upload)
        .upload_id(upload_id)
        .send()
        .await
        .unwrap();

    let data: GetObjectOutput = s3_service::download_object(&client,
&bucket_name, &key).await?;
    let data_length: u64 = data
        .content_length()
        .unwrap_or_default()
        .try_into()
        .unwrap();
    if file.metadata().unwrap().len() == data_length {
        println!("Data lengths match.");
    } else {
        println!("The data was not the same size!");
    }

    s3_service::delete_objects(&client, &bucket_name)
```

```
        .await
        .expect("Error emptying bucket.");
    s3_service::delete_bucket(&client, &bucket_name)
        .await
        .expect("Error deleting bucket.");

    Ok(())
}
```

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Hochladen eines Streams unbekannter Größe in ein Amazon S3-Objekt mithilfe eines - AWS SDK

Im folgenden Codebeispiel wird veranschaulicht, wie Sie einen Stream unbekannter Größe in ein Amazon S3-Objekt hochladen.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Verwenden Sie den AWS -CRT-basierten S3-Client.

```
import com.example.s3.util.AsyncExampleUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.async.BlockingInputStreamAsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
```

```
import java.io.ByteArrayInputStream;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;

    /**
     * @param s33CrtAsyncClient - To upload content from a stream of unknown
 size, use the AWS CRT-based S3 client. For more information, see
     *                          https://docs.aws.amazon.com/sdk-for-java/latest/
developer-guide/crt-based-s3-client.html.
     * @param bucketName - The name of the bucket.
     * @param key - The name of the object.
     * @return software.amazon.awssdk.services.s3.model.PutObjectResponse -
 Returns metadata pertaining to the put object operation.
     */
    public PutObjectResponse putObjectFromStream(S3AsyncClient s33CrtAsyncClient,
 String bucketName, String key) {

        BlockingInputStreamAsyncRequestBody body =
                AsyncRequestBody.forBlockingInputStream(null); // 'null'
 indicates a stream will be provided later.

        CompletableFuture<PutObjectResponse> responseFuture =
                s33CrtAsyncClient.putObject(r -> r.bucket(bucketName).key(key),
 body);

        // AsyncExampleUtils.randomString() returns a random string up to 100
 characters.
        String randomString = AsyncExampleUtils.randomString();
        logger.info("random string to upload: {}: length={}", randomString,
 randomString.length());

        // Provide the stream of data to be uploaded.
        body.writeInputStream(new ByteArrayInputStream(randomString.getBytes()));

        PutObjectResponse response = responseFuture.join(); // Wait for the
 response.
        logger.info("Object {} uploaded to bucket {}.", key, bucketName);
        return response;
    }
}
```

Verwenden Sie den Amazon-S3-Transfer-Manager.

```
import com.example.s3.util.AsyncExampleUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.async.BlockingInputStreamAsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedUpload;
import software.amazon.awssdk.transfer.s3.model.Upload;

import java.io.ByteArrayInputStream;
import java.util.UUID;

    /**
     * @param transferManager - To upload content from a stream of unknown size,
 use the S3TransferManager based on the AWS CRT-based S3 client.
     *                        For more information, see https://
docs.aws.amazon.com/sdk-for-java/latest/developer-guide/transfer-manager.html.
     * @param bucketName - The name of the bucket.
     * @param key - The name of the object.
     * @return - software.amazon.awssdk.transfer.s3.model.CompletedUpload - The
 result of the completed upload.
     */
    public CompletedUpload uploadStream(S3TransferManager transferManager, String
 bucketName, String key) {

        BlockingInputStreamAsyncRequestBody body =
                AsyncRequestBody.forBlockingInputStream(null); // 'null'
 indicates a stream will be provided later.

        Upload upload = transferManager.upload(builder -> builder
                .requestBody(body)
                .putObjectRequest(req -> req.bucket(bucketName).key(key))
                .build());

        // AsyncExampleUtils.randomString() returns a random string up to 100
 characters.
        String randomString = AsyncExampleUtils.randomString();
        logger.info("random string to upload: {}: length={}", randomString,
 randomString.length());

        // Provide the stream of data to be uploaded.
```

```
        body.writeInputStream(new ByteArrayInputStream(randomString.getBytes()));

        return upload.completionFuture().join();
    }
}
```

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden von Prüfsummen für die Arbeit mit einem Amazon S3-Objekt unter Verwendung eines - AWS SDK

Das folgende Codebeispiel zeigt, wie Sie Prüfsummen verwenden, um mit einem Amazon-S3-Objekt zu arbeiten.

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In den Codebeispielen wird eine Teilmenge der folgenden Importe verwendet.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ChecksumAlgorithm;
import software.amazon.awssdk.services.s3.model.ChecksumMode;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
```

```
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.RandomAccessFile;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.ByteBuffer;
import java.nio.file.Paths;
import java.security.DigestInputStream;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Base64;
import java.util.List;
import java.util.Objects;
import java.util.UUID;
```

Geben Sie einen Prüfsummenalgorithmus für die `putObject`-Methode an, wenn Sie `PutObjectRequest` erstellen.

```
    public void putObjectWithChecksum() {
        s3Client.putObject(b -> b
                .bucket(bucketName)
                .key(key)
                .checksumAlgorithm(ChecksumAlgorithm.CRC32),
                RequestBody.fromString("This is a test"));
    }
```

Überprüfen Sie die Prüfsumme für die `getObject` Methode, wenn Sie die erstellen GetObjectRequest.

```
    public GetObjectResponse getObjectWithChecksum() {
        return s3Client.getObject(b -> b
```

```
                .bucket(bucketName)
                .key(key)
                .checksumMode(ChecksumMode.ENABLED))
                .response();
    }
```

Berechnen Sie im Voraus eine Prüfsumme für die `putObject`-Methode, wenn Sie `PutObjectRequest` erstellen.

```
    public void putObjectWithPrecalculatedChecksum(String filePath) {
        String checksum = calculateChecksum(filePath, "SHA-256");

        s3Client.putObject((b -> b
                .bucket(bucketName)
                .key(key)
                .checksumSHA256(checksum)),
                RequestBody.fromFile(Paths.get(filePath)));
    }
```

Verwenden Sie den S3-Transfer-Manager zusätzlich zum AWS -CRT-basierten S3-Client, um einen mehrteiligen Upload auf transparente Weise durchzuführen, wenn die Größe des Inhalts einen Schwellenwert überschreitet. Der Standardschwellenwert beträgt 8 MB.

Sie können einen Prüfsummenalgorithmus angeben, den das SDK verwenden soll. Standardmäßig verwendet das SDK den CRC32-Algorithmus.

```
    public void multipartUploadWithChecksumTm(String filePath) {
        S3TransferManager transferManager = S3TransferManager.create();
        UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
                .putObjectRequest(b -> b
                        .bucket(bucketName)
                        .key(key)
                        .checksumAlgorithm(ChecksumAlgorithm.SHA1))
                .source(Paths.get(filePath))
                .build();
        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);
        fileUpload.completionFuture().join();
        transferManager.close();
    }
```

Verwenden Sie die S3Client-API oder (S3AsyncClient API), um einen mehrteiligen Upload durchzuführen. Wenn Sie eine zusätzliche Prüfsumme angeben, müssen Sie den Algorithmus angeben, der bei der Initiierung des Uploads verwendet werden soll. Sie müssen auch den Algorithmus für jede Teilanforderung angeben und die für jedes Teil nach dem Hochladen berechnete Prüfsumme bereitstellen.

```
   public void multipartUploadWithChecksumS3Client(String filePath) {
       ChecksumAlgorithm algorithm = ChecksumAlgorithm.CRC32;

       // Initiate the multipart upload.
       CreateMultipartUploadResponse createMultipartUploadResponse =
s3Client.createMultipartUpload(b -> b
               .bucket(bucketName)
               .key(key)
               .checksumAlgorithm(algorithm)); // Checksum specified on
initiation.
       String uploadId = createMultipartUploadResponse.uploadId();

       // Upload the parts of the file.
       int partNumber = 1;
       List<CompletedPart> completedParts = new ArrayList<>();
       ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

       try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
           long fileSize = file.length();
           int position = 0;
           while (position < fileSize) {
               file.seek(position);
               int read = file.getChannel().read(bb);

               bb.flip(); // Swap position and limit before reading from the
buffer.
               UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                       .bucket(bucketName)
                       .key(key)
                       .uploadId(uploadId)
                       .checksumAlgorithm(algorithm) // Checksum specified on
each part.
                       .partNumber(partNumber)
                       .build();

               UploadPartResponse partResponse = s3Client.uploadPart(
                       uploadPartRequest,
```

```
                        RequestBody.fromByteBuffer(bb));

                CompletedPart part = CompletedPart.builder()
                        .partNumber(partNumber)
                        .checksumCRC32(partResponse.checksumCRC32()) // Provide
the calculated checksum.
                        .eTag(partResponse.eTag())
                        .build();
                completedParts.add(part);

                bb.clear();
                position += read;
                partNumber++;
            }
        } catch (IOException e) {
            System.err.println(e.getMessage());
        }

        // Complete the multipart upload.
        s3Client.completeMultipartUpload(b -> b
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)

.multipartUpload(CompletedMultipartUpload.builder().parts(completedParts).build()));
    }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - CompleteMultipartUpload
  - CreateMultipartUpload
  - UploadPart

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Arbeiten mit versionierten Amazon S3-Objekten unter Verwendung eines - AWS SDK

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Erstellen Sie einen versionierten S3 Bucket.
- Rufen Sie alle Versionen eines Objekts ab.
- Setzen Sie eine Richtlinie auf eine frühere Version zurück.
- Löschen Sie ein versioniertes Objekt und stellen Sie es wieder her.
- Löschen Sie alle Versionen eines Objekts dauerhaft.

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie Funktionen, die S3-Aktionen umschließen.

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
 lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                   configured lifecycle rules.
```

```
    :return: The newly created bucket.
    """
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
expiration},
                    }
                ]
            }
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
```

```
        )
    except ClientError as error:
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket



def rollback_object(bucket, object_key, version_id):
    """
    Rolls back an object to an earlier version by deleting all versions that
    occurred after the specified rollback version.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that holds the object to roll back.
    :param object_key: The object to roll back.
    :param version_id: The version ID to roll back to.
    """
    # Versions must be sorted by last_modified date because delete markers are
    # at the end of the list even when they are interspersed in time.
    versions = sorted(
        bucket.object_versions.filter(Prefix=object_key),
        key=attrgetter("last_modified"),
        reverse=True,
    )

    logger.debug(
        "Got versions:\n%s",
        "\n".join(
            [
                f"\t{version.version_id}, last modified {version.last_modified}"
                for version in versions
            ]
        ),
    )

    if version_id in [ver.version_id for ver in versions]:
```

```
        print(f"Rolling back to version {version_id}")
        for version in versions:
            if version.version_id != version_id:
                version.delete()
                print(f"Deleted version {version.version_id}")
            else:
                break

        print(f"Active version is now {bucket.Object(object_key).version_id}")
    else:
        raise KeyError(
            f"{version_id} was not found in the list of versions for "
 f"{object_key}."
        )



def revive_object(bucket, object_key):
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
 marker.
    By removing the delete marker, we make the previous version the latest
 version
    and the object then presents as *not* deleted.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to revive.
    """
    # Get the latest version for the object.
    response = s3.meta.client.list_object_versions(
        Bucket=bucket.name, Prefix=object_key, MaxKeys=1
    )

    if "DeleteMarkers" in response:
        latest_version = response["DeleteMarkers"][0]
        if latest_version["IsLatest"]:
            logger.info(
                "Object %s was indeed deleted on %s. Let's revive it.",
                object_key,
```

```
                latest_version["LastModified"],
            )
            obj = bucket.Object(object_key)
            obj.Version(latest_version["VersionId"]).delete()
            logger.info(
                "Revived %s, active version is now %s  with body '%s'",
                object_key,
                obj.version_id,
                obj.get()["Body"].read(),
            )
        else:
            logger.warning(
                "Delete marker is not the latest version for %s!", object_key
            )
    elif "Versions" in response:
        logger.warning("Got an active version for %s, nothing to do.",
 object_key)
    else:
        logger.error("Couldn't get any version info for %s.", object_key)



def permanently_delete_object(bucket, object_key):
    """
    Permanently deletes a versioned object by deleting all of its versions.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to delete.
    """
    try:
        bucket.object_versions.filter(Prefix=object_key).delete()
        logger.info("Permanently deleted all versions of object %s.", object_key)
    except ClientError:
        logger.exception("Couldn't delete all versions of %s.", object_key)
        raise
```

Laden Sie die Strophe eines Gedichts in ein versioniertes Objekt hoch und führen Sie eine Reihe von Aktionen dafür aus.

```
def usage_demo_single_object(obj_prefix="demo-versioning/"):
    """
    Demonstrates usage of versioned object functions. This demo uploads a stanza
    of a poem and performs a series of revisions, deletions, and revivals on it.

    :param obj_prefix: The prefix to assign to objects created by this demo.
    """
    with open("father_william.txt") as file:
        stanzas = file.read().split("\n\n")

    width = get_terminal_size((80, 20))[0]
    print("-" * width)
    print("Welcome to the usage demonstration of Amazon S3 versioning.")
    print(
        "This demonstration uploads a single stanza of a poem to an Amazon "
        "S3 bucket and then applies various revisions to it."
    )
    print("-" * width)
    print("Creating a version-enabled bucket for the demo...")
    bucket = create_versioned_bucket("bucket-" + str(uuid.uuid1()), obj_prefix)

    print("\nThe initial version of our stanza:")
    print(stanzas[0])

    # Add the first stanza and revise it a few times.
    print("\nApplying some revisions to the stanza...")
    obj_stanza_1 = bucket.Object(f"{obj_prefix}stanza-1")
    obj_stanza_1.put(Body=bytes(stanzas[0], "utf-8"))
    obj_stanza_1.put(Body=bytes(stanzas[0].upper(), "utf-8"))
    obj_stanza_1.put(Body=bytes(stanzas[0].lower(), "utf-8"))
    obj_stanza_1.put(Body=bytes(stanzas[0][::-1], "utf-8"))
    print(
        "The latest version of the stanza is now:",
        obj_stanza_1.get()["Body"].read().decode("utf-8"),
        sep="\n",
    )

    # Versions are returned in order, most recent first.
    obj_stanza_1_versions =
 bucket.object_versions.filter(Prefix=obj_stanza_1.key)
```

```
    print(
        "The version data of the stanza revisions:",
        *[
            f"    {version.version_id}, last modified {version.last_modified}"
            for version in obj_stanza_1_versions
        ],
        sep="\n",
    )

    # Rollback two versions.
    print("\nRolling back two versions...")
    rollback_object(bucket, obj_stanza_1.key, list(obj_stanza_1_versions)
[2].version_id)
    print(
        "The latest version of the stanza:",
        obj_stanza_1.get()["Body"].read().decode("utf-8"),
        sep="\n",
    )

    # Delete the stanza
    print("\nDeleting the stanza...")
    obj_stanza_1.delete()
    try:
        obj_stanza_1.get()
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchKey":
            print("The stanza is now deleted (as expected).")
        else:
            raise

    # Revive the stanza
    print("\nRestoring the stanza...")
    revive_object(bucket, obj_stanza_1.key)
    print(
        "The stanza is restored! The latest version is again:",
        obj_stanza_1.get()["Body"].read().decode("utf-8"),
        sep="\n",
    )

    # Permanently delete all versions of the object. This cannot be undone!
    print("\nPermanently deleting all versions of the stanza...")
    permanently_delete_object(bucket, obj_stanza_1.key)
    obj_stanza_1_versions =
 bucket.object_versions.filter(Prefix=obj_stanza_1.key)
```

```
    if len(list(obj_stanza_1_versions)) == 0:
        print("The stanza has been permanently deleted and now has no versions.")
    else:
        print("Something went wrong. The stanza still exists!")

    print(f"\nRemoving {bucket.name}...")
    bucket.delete()
    print(f"{bucket.name} deleted.")
    print("Demo done!")
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - CreateBucket
  - DeleteObject
  - ListObjectVersions
  - PutBucketLifecycleConfiguration

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Serverless-Beispiele für Amazon S3 unter Verwendung von AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie Amazon S3 mit - AWS SDKs verwenden.

Beispiele

- Aufrufen einer Lambda-Funktion über einen Amazon-S3-Auslöser

Aufrufen einer Lambda-Funktion über einen Amazon-S3-Auslöser

In den folgenden Codebeispiele wird die Implementierung einer Lambda-Funktion gezeigt, die ein Ereignis empfängt, das durch Hochladen eines Objekts in einen S3-Bucket ausgelöst wird. Die

Funktion ruft den Namen des S3-Buckets sowie den Objektschlüssel aus dem Ereignisparameter ab und ruft die Amazon-S3-API auf, um den Inhaltstyp des Objekts abzurufen und zu protokollieren.

.NET

AWS SDK for .NET

> **Note**
>
> Auf gibt es mehr GitHub. Das vollständige Beispiel sowie eine Anleitung zum Einrichten und Ausführen finden Sie im Repository mit Serverless-Beispielen.

Nutzen eines S3-Ereignisses mit Lambda unter Verwendung von .NET

```
using System.Threading.Tasks;
using Amazon.Lambda.Core;
using Amazon.S3;
using System;
using Amazon.Lambda.S3Events;
using System.Web;

// Assembly attribute to enable the Lambda function's JSON input to be converted
 into a .NET class.
[assembly:
 LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambdaJsonSeri

namespace S3Integration
{
    public class Function
    {
        private static AmazonS3Client _s3Client;
        public Function() : this(null)
        {
        }

        internal Function(AmazonS3Client s3Client)
        {
            _s3Client = s3Client ?? new AmazonS3Client();
        }

        public async Task<string> Handler(S3Event evt, ILambdaContext context)
```

```
        {
            try
            {
                if (evt.Records.Count <= 0)
                {
                    context.Logger.LogLine("Empty S3 Event received");
                    return string.Empty;
                }

                var bucket = evt.Records[0].S3.Bucket.Name;
                var key = HttpUtility.UrlDecode(evt.Records[0].S3.Object.Key);

                context.Logger.LogLine($"Request is for {bucket} and {key}");

                var objectResult = await _s3Client.GetObjectAsync(bucket, key);

                context.Logger.LogLine($"Returning {objectResult.Key}");

                return objectResult.Key;
            }
            catch (Exception e)
            {
                context.Logger.LogLine($"Error processing request -
 {e.Message}");

                return string.Empty;
            }
        }
    }
}
```

Go

SDK für Go V2

> **Note**
>
> Auf gibt es mehr GitHub. Das vollständige Beispiel sowie eine Anleitung zum Einrichten und Ausführen finden Sie im Repository mit Serverless-Beispielen.

Nutzen eines S3-Ereignisses mit Lambda unter Verwendung von Go

```
package main

import (
 "context"
 "log"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/s3"
)

func handler(ctx context.Context, s3Event events.S3Event) error {
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Printf("failed to load default config: %s", err)
  return err
 }
 s3Client := s3.NewFromConfig(sdkConfig)

 for _, record := range s3Event.Records {
  bucket := record.S3.Bucket.Name
  key := record.S3.Object.URLDecodedKey
  headOutput, err := s3Client.HeadObject(ctx, &s3.HeadObjectInput{
   Bucket: &bucket,
   Key:    &key,
  })
  if err != nil {
   log.Printf("error getting head of object %s/%s: %s", bucket, key, err)
   return err
  }
  log.Printf("successfully retrieved %s/%s of type %s", bucket, key,
 *headOutput.ContentType)
 }

 return nil
}

func main() {
 lambda.Start(handler)
}
```

Java

SDK für Java 2.x

> **Note**
>
> Auf gibt es mehr GitHub. Das vollständige Beispiel sowie eine Anleitung zum Einrichten und Ausführen finden Sie im Repository mit Serverless-Beispielen.

Nutzen eines S3-Ereignisses mit Lambda unter Verwendung von Java

```
package example;

import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.S3Client;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.S3Event;
import
 com.amazonaws.services.lambda.runtime.events.models.s3.S3EventNotification.S3EventNotifi

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class Handler implements RequestHandler<S3Event, String> {
    private static final Logger logger = LoggerFactory.getLogger(Handler.class);
    @Override
    public String handleRequest(S3Event s3event, Context context) {
        try {
          S3EventNotificationRecord record = s3event.getRecords().get(0);
          String srcBucket = record.getS3().getBucket().getName();
          String srcKey = record.getS3().getObject().getUrlDecodedKey();

          S3Client s3Client = S3Client.builder().build();
          HeadObjectResponse headObject = getHeadObject(s3Client, srcBucket,
 srcKey);

          logger.info("Successfully retrieved " + srcBucket + "/" + srcKey + " of
 type " + headObject.contentType());
```

```
            return "Ok";
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    private HeadObjectResponse getHeadObject(S3Client s3Client, String bucket,
 String key) {
        HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
                .bucket(bucket)
                .key(key)
                .build();
        return s3Client.headObject(headObjectRequest);
    }
}
```

JavaScript

SDK für JavaScript (v2)

> **Note**
>
> Auf gibt es mehr GitHub. Das vollständige Beispiel sowie eine Anleitung zum Einrichten und Ausführen finden Sie im Repository mit Serverless-Beispielen.

Nutzen eines S3-Ereignisses mit Lambda unter Verwendung von JavaScript.

```
const aws = require('aws-sdk');

const s3 = new aws.S3({ apiVersion: '2006-03-01' });

exports.handler = async (event, context) => {
    // Get the object from the event and show its content type
    const bucket = event.Records[0].s3.bucket.name;
    const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g,
 ' '));
    const params = {
        Bucket: bucket,
        Key: key,
    };
    try {
```

```
        const { ContentType } = await s3.headObject(params).promise();
        console.log('CONTENT TYPE:', ContentType);
        return ContentType;
    } catch (err) {
        console.log(err);
        const message = `Error getting object ${key} from bucket ${bucket}. Make
 sure they exist and your bucket is in the same region as this function.`;
        console.log(message);
        throw new Error(message);
    }
};
```

Nutzen eines S3-Ereignisses mit Lambda unter Verwendung von TypeScript.

```
import { S3Event } from 'aws-lambda';
import { S3Client, HeadObjectCommand } from '@aws-sdk/client-s3';

const s3 = new S3Client({ region: process.env.AWS_REGION });

export const handler = async (event: S3Event): Promise<string | undefined> => {
  // Get the object from the event and show its content type
  const bucket = event.Records[0].s3.bucket.name;
  const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, '
 '));
  const params = {
    Bucket: bucket,
    Key: key,
  };
  try {
    const { ContentType } = await s3.send(new HeadObjectCommand(params));
    console.log('CONTENT TYPE:', ContentType);
    return ContentType;
  } catch (err) {
    console.log(err);
    const message = `Error getting object ${key} from bucket ${bucket}. Make sure
 they exist and your bucket is in the same region as this function.`;
    console.log(message);
    throw new Error(message);
  }
};
```

Python

SDK für Python (Boto3)

> **Note**
>
> Auf gibt es mehr GitHub. Das vollständige Beispiel sowie eine Anleitung zum Einrichten und Ausführen finden Sie im Repository mit Serverless-Beispielen.

Nutzen eines S3-Ereignisses mit Lambda unter Verwendung von Python

```
import json
import urllib.parse
import boto3

print('Loading function')

s3 = boto3.client('s3')


def lambda_handler(event, context):
    #print("Received event: " + json.dumps(event, indent=2))

    # Get the object from the event and show its content type
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'],
 encoding='utf-8')
    try:
        response = s3.get_object(Bucket=bucket, Key=key)
        print("CONTENT TYPE: " + response['ContentType'])
        return response['ContentType']
    except Exception as e:
        print(e)
        print('Error getting object {} from bucket {}. Make sure they exist and
 your bucket is in the same region as this function.'.format(key, bucket))
        raise e
```

Rust

SDK für Rust

> **Note**
>
> Auf gibt es mehr GitHub. Das vollständige Beispiel sowie eine Anleitung zum Einrichten und Ausführen finden Sie im Repository mit Serverless-Beispielen.

Nutzen eines S3-Ereignisses mit Lambda unter Verwendung von Rust

```
use aws_lambda_events::event::s3::S3Event;
use aws_sdk_s3::{Client};
use lambda_runtime::{run, service_fn, Error, LambdaEvent};


/// Main function
#[tokio::main]
async fn main() -> Result<(), Error> {
    tracing_subscriber::fmt()
        .with_max_level(tracing::Level::INFO)
        .with_target(false)
        .without_time()
        .init();

    // Initialize the AWS SDK for Rust
    let config = aws_config::load_from_env().await;
    let s3_client = Client::new(&config);

    let res = run(service_fn(|request: LambdaEvent<S3Event>| {
        function_handler(&s3_client, request)
    })).await;

    res
}

async fn function_handler(
    s3_client: &Client,
    evt: LambdaEvent<S3Event>
) -> Result<(), Error> {
    tracing::info!(records = ?evt.payload.records.len(), "Received request from
 SQS");
```

```
    if evt.payload.records.len() == 0 {
        tracing::info!("Empty S3 event received");
    }

    let bucket = evt.payload.records[0].s3.bucket.name.as_ref().expect("Bucket
 name to exist");
    let key = evt.payload.records[0].s3.object.key.as_ref().expect("Object key to
 exist");

    tracing::info!("Request is for {} and object {}", bucket, key);

    let s3_get_object_result = s3_client
        .get_object()
        .bucket(bucket)
        .key(key)
        .send()
        .await;

    match s3_get_object_result {
        Ok(_) => tracing::info!("S3 Get Object success, the s3GetObjectResult
 contains a 'body' property of type ByteStream"),
        Err(_) => tracing::info!("Failure with S3 Get Object request")
    }

    Ok(())
}
```

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Serviceübergreifende Beispiele für Amazon S3 unter Verwendung von AWS SDKs

Die folgenden Beispielanwendungen verwenden - AWS SDKs, um Amazon S3 mit anderen zu kombinieren AWS-Services. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen der Anwendung finden.

Beispiele

- Eine Amazon-Transcribe-App entwickeln
- Konvertieren von Text in Sprache und zurück in Text mithilfe eines - AWS SDK
- Eine Anwendung für Foto-Asset-Management erstellen, mit der Benutzer Fotos mithilfe von Labels verwalten können
- Erstellen Sie eine Amazon-Textract-Explorer-Anwendung
- Erkennen von PSA in Bildern mit Amazon Rekognition mithilfe eines - AWS SDK
- Erkennen von Entitäten in Text, der mithilfe eines - AWS SDK aus einem Bild extrahiert wurde
- Erkennen von Gesichtern in einem Bild mithilfe eines - AWS SDK
- Erkennen von Objekten in Bildern mit Amazon Rekognition mithilfe eines AWS -SDK
- Erkennen von Personen und Objekten in einem Video mit Amazon Rekognition mithilfe eines AWS -SDK
- EXIF- und andere Image-Informationen mit einem - AWS SDK speichern

Eine Amazon-Transcribe-App entwickeln

Das folgende Codebeispiel zeigt, wie Amazon Transcribe verwendet wird, um Sprachaufnahmen im Browser zu transkribieren und anzuzeigen.

JavaScript

SDK für JavaScript (v3)

Erstellen Sie eine App, die Amazon Transcribe verwendet, um Sprachaufnahmen im Browser zu transkribieren und anzuzeigen. Die App verwendet zwei Amazon Simple Storage Service (Amazon S3)-Buckets, einen zum Hosten des Anwendungscodes und einen zum Speichern von Transkriptionen. Die App verwendet einen Amazon-Cognito-Benutzerpool zur Authentifizierung Ihrer Benutzer. Authentifizierte Benutzer verfügen über AWS Identity and Access Management (IAM)-Berechtigungen für den Zugriff auf die erforderlichen AWS Services.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Dieses Beispiel ist auch verfügbar im AWS SDK for JavaScript Entwicklerhandbuch für v3.

In diesem Beispiel verwendete Dienste

- Amazon Cognito Identity

- Amazon S3
- Amazon Transcribe

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Konvertieren von Text in Sprache und zurück in Text mithilfe eines - AWS SDK

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Verwenden Sie Amazon Polly, um eine Nur-Text-Eingabedatei (UTF-8) in eine Audiodatei zu synthetisieren.
- Laden Sie die Audiodatei in einen Amazon-S3-Bucket hoch.
- Konvertieren Sie die Audiodatei mit Amazon Transcribe in Text.
- Zeigen Sie den Text an.

Rust

SDK für Rust

Verwenden Sie Amazon Polly, um eine Klartext-Eingabedatei (UTF-8) in eine Audiodatei zu synthetisieren, die Audiodatei in einen Amazon-S3-Bucket hochzuladen, diese Audiodatei mit Amazon Transcribe in Text zu konvertieren und den Text anzuzeigen.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Polly
- Amazon S3
- Amazon Transcribe

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Eine Anwendung für Foto-Asset-Management erstellen, mit der Benutzer Fotos mithilfe von Labels verwalten können

Die folgenden Codebeispiele zeigen, wie eine Serverless-Anwendung erstellt wird, mit der Benutzer Fotos mithilfe von Labels verwalten können.

.NET

AWS SDK for .NET

Zeigt, wie eine Anwendung zur Verwaltung von Fotobeständen entwickelt wird, die mithilfe von Amazon Rekognition Labels in Bildern erkennt und sie für einen späteren Abruf speichert.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Einen tiefen Einblick in den Ursprung dieses Beispiels finden Sie im Beitrag in der AWS - Community.

In diesem Beispiel verwendete Dienste

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

C++

SDK für C++

Zeigt, wie eine Anwendung zur Verwaltung von Fotobeständen entwickelt wird, die mithilfe von Amazon Rekognition Labels in Bildern erkennt und sie für einen späteren Abruf speichert.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Einen tiefen Einblick in den Ursprung dieses Beispiels finden Sie im Beitrag in der AWS - Community.

In diesem Beispiel verwendete Dienste

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Java

SDK für Java 2.x

Zeigt, wie eine Anwendung zur Verwaltung von Fotobeständen entwickelt wird, die mithilfe von Amazon Rekognition Labels in Bildern erkennt und sie für einen späteren Abruf speichert.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Einen tiefen Einblick in den Ursprung dieses Beispiels finden Sie im Beitrag in der AWS - Community.

In diesem Beispiel verwendete Dienste

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

JavaScript

SDK für JavaScript (v3)

Zeigt, wie eine Anwendung zur Verwaltung von Fotobeständen entwickelt wird, die mithilfe von Amazon Rekognition Labels in Bildern erkennt und sie für einen späteren Abruf speichert.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Einen tiefen Einblick in den Ursprung dieses Beispiels finden Sie im Beitrag in der AWS - Community.

In diesem Beispiel verwendete Dienste

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Kotlin

SDK für Kotlin

Zeigt, wie eine Anwendung zur Verwaltung von Fotobeständen entwickelt wird, die mithilfe von Amazon Rekognition Labels in Bildern erkennt und sie für einen späteren Abruf speichert.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Einen tiefen Einblick in den Ursprung dieses Beispiels finden Sie im Beitrag in der AWS - Community.

In diesem Beispiel verwendete Dienste

- API Gateway
- DynamoDB

- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

PHP

SDK für PHP

Zeigt, wie eine Anwendung zur Verwaltung von Fotobeständen entwickelt wird, die mithilfe von Amazon Rekognition Labels in Bildern erkennt und sie für einen späteren Abruf speichert.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Einen tiefen Einblick in den Ursprung dieses Beispiels finden Sie im Beitrag in der AWS - Community.

In diesem Beispiel verwendete Dienste

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Rust

SDK für Rust

Zeigt, wie eine Anwendung zur Verwaltung von Fotobeständen entwickelt wird, die mithilfe von Amazon Rekognition Labels in Bildern erkennt und sie für einen späteren Abruf speichert.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

Einen tiefen Einblick in den Ursprung dieses Beispiels finden Sie im Beitrag in der AWS - Community.

In diesem Beispiel verwendete Dienste

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen Sie eine Amazon-Textract-Explorer-Anwendung

Die folgenden Code-Beispiele zeigen, wie man die Amazon-Textract-Ausgabe in einer interaktiven Anwendung untersuchen kann.

JavaScript

SDK für JavaScript (v3)

Zeigt, wie Sie mit eine React-Anwendung AWS SDK for JavaScript erstellen, die Amazon Textract verwendet, um Daten aus einem Dokumentbild zu extrahieren und auf einer interaktiven Webseite anzuzeigen. Dieses Beispiel wird in einem Webbrowser ausgeführt und erfordert eine authentifizierte Amazon-Cognito-Identität für Anmeldeinformationen. Es verwendet Amazon Simple Storage Service (Amazon S3) zur Speicherung und fragt für Benachrichtigungen eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange ab, die ein Amazon Simple Notification Service (Amazon SNS)-Thema abonniert hat.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Cognito Identity

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Python

SDK für Python (Boto3)

Zeigt, wie Sie AWS SDK for Python (Boto3) mit Amazon Textract verwenden, um Text-, Formular- und Tabellenelemente in einem Dokumentbild zu erkennen. Das Eingabe-Image und die Amazon-Textract-Ausgabe werden in einer Tkinter-Anwendung angezeigt, mit der Sie die erkannten Elemente untersuchen können.

- Senden Sie ein Dokument-Image an Amazon Textract und untersuchen Sie die Ausgabe erkannter Elemente.
- Senden Sie Images direkt an Amazon Textract oder über einen Amazon Simple Storage Service (Amazon S3)-Bucket.
- Verwenden Sie asynchrone APIs, um einen Auftrag zu starten, der eine Benachrichtigung an ein Amazon Simple Notification Service (Amazon SNS)-Thema veröffentlicht.
- Stellen Sie eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange ab, um eine Meldung zum Abschluss des Auftrags zu erhalten.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erkennen von PSA in Bildern mit Amazon Rekognition mithilfe eines - AWS SDK

Die folgenden Code-Beispiele zeigen, wie man eine App erstellt, die Amazon Rekognition verwendet, um Persönliche Schutzausrüstung (PSA) in Bildern zu erkennen.

Java

SDK für Java 2.x

Zeigt, wie Sie eine - AWS Lambda Funktion erstellen, die Bilder mit persönlicher Schutzausrüstung erkennt.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- DynamoDB
- Amazon Rekognition
- Amazon S3
- Amazon SES

JavaScript

SDK für JavaScript (v3)

Zeigt, wie man Amazon Rekognition mit der verwendet, AWS SDK for JavaScript um eine Anwendung zur Erkennung persönlicher Schutzausrüstung (PSA) in Bildern zu erstellen, die sich in einem Amazon Simple Storage Service (Amazon S3)-Bucket befinden. Die App speichert die Ergebnisse in einer Amazon-DynamoDB-Tabelle und sendet dem Administrator eine E-Mail-Benachrichtigung mit den Ergebnissen über Amazon Simple Email Service (Amazon SES).

So funktioniert es:

- Erstellen Sie mit Amazon Cognito einen nicht authentifizierten Benutzer.
- Analysieren Sie mit Amazon Rekognition Bilder für PSA.
- Verifizieren Sie eine E-Mail-Adresse für Amazon SES.

- Aktualisieren Sie eine DynamoDB-Tabelle mit Ergebnissen.
- Senden Sie eine E-Mail-Benachrichtigung mit Amazon SES.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- DynamoDB
- Amazon Rekognition
- Amazon S3
- Amazon SES

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erkennen von Entitäten in Text, der mithilfe eines - AWS SDK aus einem Bild extrahiert wurde

Das folgende Codebeispiel zeigt, wie Amazon Comprehend verwendet wird, um Entitäten in Text zu erkennen, der von Amazon Textract aus einem in Amazon S3 gespeicherten Bild extrahiert wurde.

Python

SDK für Python (Boto3)

Zeigt, wie Sie die AWS SDK for Python (Boto3) in einem Jupyter-Notebook verwenden, um Entitäten in Text zu erkennen, der aus einem Bild extrahiert wird. In diesem Beispiel extrahiert Amazon Textract Text aus einem Bild, das in Amazon Simple Storage Service (Amazon S3) und Amazon Comprehend gespeichert ist, um Entitäten im extrahierten Text zu erkennen.

Dieses Beispiel ist ein Jupyter Notebook und muss in einer Umgebung ausgeführt werden, die Notebooks hosten kann. Anweisungen zum Ausführen des Beispiels mit Amazon SageMakerfinden Sie in den Anweisungen unter TextractAndComprehendNotebook.ipynb .

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Comprehend
- Amazon S3
- Amazon Textract

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erkennen von Gesichtern in einem Bild mithilfe eines - AWS SDK

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Speichern Sie ein Bild in einem Amazon-S3-Bucket.
- Verwenden Sie Amazon Rekognition, um Gesichtsdetails wie Altersgruppe, Geschlecht und Emotionen (z B. Lächeln) zu erkennen.
- Zeigen Sie diese Details an.

Rust

SDK für Rust

Speichern Sie das Bild in einem Amazon-S3-Bucket mit einem uploads-Präfix, verwenden Sie Amazon Rekognition, um Gesichtsdetails wie Altersgruppe, Geschlecht und Emotionen (Lächeln usw.) zu erkennen, und zeigen Sie diese Details an.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition
- Amazon S3

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erkennen von Objekten in Bildern mit Amazon Rekognition mithilfe eines AWS -SDK

Die folgenden Code-Beispiele zeigen, wie man eine App erstellt, die Amazon Rekognition verwendet, um Objekte nach Kategorien in Bildern zu erkennen.

.NET

AWS SDK for .NET

Zeigt, wie Sie die Amazon-Rekognition-.NET-API verwenden, um eine App zu erstellen, die Amazon Rekognition verwendet, um Objekte nach Kategorien in Bildern zu identifizieren, die sich in einem Bucket von Amazon Simple Storage Service (Amazon S3) befinden. Die App sendet dem Administrator eine E-Mail-Benachrichtigung mit den Ergebnissen über Amazon Simple Email Service (Amazon SES).

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition
- Amazon S3
- Amazon SES

Java

SDK für Java 2.x

Zeigt, wie man die Amazon-Rekognition-Java-API verwendet, um eine App zu erstellen, die Amazon Rekognition verwendet, um Objekte nach Kategorien in Bildern zu identifizieren, die sich in einem Amazon Simple Storage Service (Amazon S3)-Bucket befinden. Die App sendet dem Administrator eine E-Mail-Benachrichtigung mit den Ergebnissen über Amazon Simple Email Service (Amazon SES).

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition

- Amazon S3
- Amazon SES

JavaScript

SDK für JavaScript (v3)

Zeigt, wie man Amazon Rekognition mit verwendet, AWS SDK for JavaScript um eine App zu erstellen, die Amazon Rekognition verwendet, um Objekte nach Kategorien in Bildern zu identifizieren, die sich in einem Amazon Simple Storage Service (Amazon S3)-Bucket befinden. Die App sendet dem Administrator eine E-Mail-Benachrichtigung mit den Ergebnissen über Amazon Simple Email Service (Amazon SES).

So funktioniert es:

- Erstellen Sie mit Amazon Cognito einen nicht authentifizierten Benutzer.
- Analysieren Sie mit Amazon Rekognition Bilder für Objekte.
- Verifizieren Sie eine E-Mail-Adresse für Amazon SES.
- Senden Sie eine E-Mail-Benachrichtigung mit Amazon SES.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition
- Amazon S3
- Amazon SES

Kotlin

SDK für Kotlin

Zeigt, wie man die Amazon-Rekognition-Kotlin-API verwendet, um eine App zu erstellen, die Amazon Rekognition verwendet, um Objekte nach Kategorien in Bildern zu identifizieren, die sich in einem Amazon Simple Storage Service (Amazon S3)-Bucket befinden. Die App sendet dem Administrator eine E-Mail-Benachrichtigung mit den Ergebnissen über Amazon Simple Email Service (Amazon SES).

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition
- Amazon S3
- Amazon SES

Python

SDK für Python (Boto3)

Zeigt Ihnen, wie Sie mit AWS SDK for Python (Boto3) eine Webanwendung erstellen, mit der Sie Folgendes tun können:

- Laden Sie Fotos in einen Bucket von Amazon Simple Storage Service (Amazon S3) hoch.
- Verwenden Sie Amazon Rekognition, um die Fotos zu analysieren und zu markieren.
- Verwenden Sie Amazon Simple Email Service (Amazon SES), um E-Mail-Berichte von Bildanalysen zu senden.

Dieses Beispiel enthält zwei Hauptkomponenten: eine in geschriebene Webseite JavaScript , die mit React erstellt wurde, und einen in Python geschriebenen REST-Service, der mit Flask-RESTful erstellt wurde.

Sie können die React-Webseite verwenden, um Folgendes auszuführen:

- Zeigen Sie eine Liste der Bilder an, die in Ihrem S3-Bucket gespeichert sind.
- Laden Sie Bilder von Ihrem Computer in Ihren S3-Bucket hoch.
- Zeigen Sie Bilder und Markierungen an, die Elemente identifizieren, welche im Bild erkannt werden.
- Rufen Sie einen Bericht über alle Bilder in Ihrem S3-Bucket ab und senden Sie eine E-Mail mit dem Bericht.

Die Webseite ruft den REST-Service auf. Der Service sendet Anforderungen an AWS , um die folgenden Aktionen durchzuführen:

- Die Liste der Bilder abrufen und in Ihrem S3-Bucket filtern.
- Fotos in Ihren S3-Bucket hochladen.

- Verwenden Sie Amazon Rekognition, um einzelne Fotos zu analysieren und eine Liste von Markierungen zu erhalten, die die auf dem Foto erkannten Elemente identifizieren.
- Analysieren Sie alle Fotos in Ihrem S3-Bucket und verwenden Sie Amazon SES, um einen Bericht per E-Mail zu senden.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition
- Amazon S3
- Amazon SES

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erkennen von Personen und Objekten in einem Video mit Amazon Rekognition mithilfe eines AWS -SDK

Die folgenden Code-Beispiele zeigen, wie man Personen und Objekte in einem Video mit Amazon Rekognition erkennt.

Java

SDK für Java 2.x

Zeigt, wie man die Amazon-Rekognition-Java-API verwendet, um eine App zu erstellen, die Gesichter und Objekte in Videos erkennt, die sich in einem Amazon Simple Storage Service (Amazon S3)-Bucket befinden. Die App sendet dem Administrator eine E-Mail-Benachrichtigung mit den Ergebnissen über Amazon Simple Email Service (Amazon SES).

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition

- Amazon S3
- Amazon SES

JavaScript

SDK für JavaScript (v3)

Zeigt, wie man Amazon Rekognition mit verwendet AWS SDK for JavaScript , um eine App zu erstellen, die Gesichter und Objekte in Videos erkennt, die sich in einem Amazon Simple Storage Service (Amazon S3)-Bucket befinden. Die App sendet dem Administrator eine E-Mail-Benachrichtigung mit den Ergebnissen über Amazon Simple Email Service (Amazon SES).

So funktioniert es:

- Erstellen Sie mit Amazon Cognito einen nicht authentifizierten Benutzer.
- Analysieren Sie mit Amazon Rekognition Bilder für PSA.
- Verifizieren Sie eine E-Mail-Adresse für Amazon SES.
- Senden Sie eine E-Mail-Benachrichtigung mit Amazon SES.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Rekognition
- Amazon S3
- Amazon SES

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

EXIF- und andere Image-Informationen mit einem - AWS SDK speichern

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Rufen Sie EXIF-Informationen aus einer JPG-, JPEG- oder PNG-Datei ab.
- Laden Sie die Bilddatei in einen Amazon-S3-Bucket hoch.

- Verwenden Sie Amazon Rekognition, um die drei wichtigsten Attribute (Labels) in der Datei zu identifizieren.
- Fügen Sie die EXIF- und Label-Informationen einer Amazon-DynamoDB-Tabelle in der Region hinzu.

Rust

SDK für Rust

Rufen Sie EXIF-Informationen aus einer JPG-, JPEG- oder PNG-Datei ab, laden Sie die Bilddatei in einen Amazon-S3-Bucket hoch und identifizieren Sie mit Amazon Rekognition die drei wichtigsten Attribute (Labels in Amazon Rekognition) in der Datei. Fügen Sie die EXIF- und Labelinformationen dann einer Amazon-DynamoDB-Tabelle in der Region hinzu.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel auf GitHub.

In diesem Beispiel verwendete Dienste

- DynamoDB
- Amazon Rekognition
- Amazon S3

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter Verwenden dieses Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Fehlerbehebung

Dieser Abschnitt beschreibt die Fehlerbehebung bei Amazon-S3-Funktionen und erläutert, wie Sie die bei der Kontaktaufnahme mit dem AWS Support benötigten Anfrage-IDs erhalten.

Themen

Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3

Die folgenden Themen behandeln die häufigsten Ursachen von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3.

Themen

> **Note**
>
> Wenn Sie versuchen, ein Problem mit Berechtigungen zu beheben, beginnen Sie mit dem Abschnitt Bucket-Richtlinien und IAM-Richtlinien und befolgen Sie die Anweisungen unter Tipps zum Überprüfen von Berechtigungen.

Bucket-Richtlinien und IAM-Richtlinien

Vorgänge auf Bucket-Ebene

Wenn es keine Bucket-Richtlinie gibt, lässt der Bucket implizit Anforderungen von jeder AWS Identity and Access Management (IAM)-Identität in dem Konto zu, das im Besitz des Buckets ist. Ebenso lehnt der Bucket implizit Anforderungen von anderen IAM-Identitäten von anderen Konten sowie anonyme (unsignierte) Anforderungen ab. Wenn jedoch keine IAM-Benutzerrichtlinie vorhanden ist, wird der Anforderer (sofern es sich nicht um den Root-Benutzer handelt) implizit daran gehindert, Anforderungen zu stellen. Weitere Informationen zu dieser Evaluierungslogik finden Sie unter Ermitteln, ob eine Anforderung innerhalb eines Kontos zugelassen oder verweigert wird im IAM-Benutzerhandbuch.

Vorgänge auf Objektebene

Wenn das Objekt dem Konto gehört, das den Bucket besitzt, funktionieren die Bucket-Richtlinie und die IAM-Benutzerrichtlinie für Vorgänge auf Objektebene genauso wie für Vorgänge auf Bucket-Ebene. Wenn es beispielsweise keine Bucket-Richtlinie gibt, lässt der Bucket implizit Objektanforderungen von jeder IAM-Identität in dem Konto zu, das im Besitz des Buckets ist. Ebenso lehnt der Bucket implizit Objektanforderungen von anderen IAM-Identitäten von anderen Konten sowie anonyme (unsignierte) Anforderungen ab. Wenn jedoch keine IAM-Benutzerrichtlinie vorhanden ist, wird der Anforderer (sofern es sich nicht um den Root-Benutzer handelt) implizit daran gehindert, Objektanforderungen zu stellen.

Wenn das Objekt einem externen Konto gehört, kann der Zugriff auf das Objekt nur über Objekt-Zugriffssteuerungslisten (ACLs) gewährt werden. Die Bucket-Richtlinie und die IAM-Benutzerrichtlinie können weiterhin verwendet werden, um Objektanforderungen zu verweigern.

Stellen Sie daher sicher, dass die folgenden Voraussetzungen erfüllt sind, um sicherzugehen, dass Ihre Bucket-Richtlinie oder IAM-Benutzerrichtlinie keinen Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) verursacht:

- Für den Zugriff auf dasselbe Konto darf es weder in der Bucket-Richtlinie noch in der IAM-Benutzerrichtlinie eine explizite `Deny`-Anweisung gegen den Anforderer geben, dem Sie Berechtigungen gewähren möchten. Wenn Sie Berechtigungen nur mithilfe der Bucket-Richtlinie und der IAM-Benutzerrichtlinie gewähren möchten, muss eine dieser Richtlinien mindestens eine explizite `Allow`-Anweisung enthalten.
- Für den kontoübergreifenden Zugriff darf es weder in der Bucket-Richtlinie noch in der IAM-Benutzerrichtlinie eine explizite `Deny`-Anweisung gegen den Anforderer geben, dem Sie Berechtigungen gewähren möchten. Wenn Sie kontoübergreifende Berechtigungen nur mithilfe der Bucket-Richtlinie und der IAM-Benutzerrichtlinie gewähren möchten, müssen sowohl die Bucket-Richtlinie als auch die IAM-Benutzerrichtlinie des Anforderers eine explizite `Allow`-Anweisung enthalten.

> **Note**
>
> `Allow`-Anweisungen in einer Bucket-Richtlinie gelten nur für Objekte, die demselben Konto gehören, das im Besitz des Buckets ist. `Deny`-Anweisungen in einer Bucket-Richtlinie gelten jedoch für alle Objekte, unabhängig von der Objekteigentümerschaft.

So überprüfen oder bearbeiten Sie Ihre Bucket-Richtlinie

> **Note**
>
> Um eine Bucket-Richtlinie anzuzeigen oder zu bearbeiten, benötigen Sie die Berechtigung `s3:GetBucketPolicy`.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, für den Sie eine Bucket-Richtlinie anzeigen oder bearbeiten möchten.
4. Wählen Sie die Registerkarte Berechtigungen.
5. Wählen Sie unter Bucket-Richtlinie Bearbeiten aus. Die Seite Bucket-Richtlinie bearbeiten wird angezeigt.

Verwenden Sie den Befehl get-bucket-policy, um Ihre Bucket-Richtlinie über die AWS Command Line Interface (AWS CLI) zu überprüfen oder zu bearbeiten.

> **Note**
>
> Wenn Sie aufgrund einer falschen Bucket-Richtlinie von einem Bucket ausgeschlossen werden, melden Sie sich mit Ihren Root-Benutzer-Anmeldeinformationen bei der AWS Management Console an. Um wieder Zugriff auf Ihren Bucket zu erhalten, löschen Sie die Bucket-Richtlinie unter Verwendung Ihren Root-Benutzer-Anmeldeinformationen.

Tipps zum Überprüfen von Berechtigungen

Gehen Sie wie folgt vor, um zu überprüfen, ob der Anforderer über die erforderlichen Berechtigungen für die Ausführung einer Amazon-S3-Operation verfügt:

- Ermitteln Sie den Anforderer. Bei einer unsignierten Anforderung handelt es sich um eine anonyme Anforderung ohne IAM-Benutzerrichtlinie. Bei einer Anforderung mit einer vorsignierten URL entspricht die Benutzerrichtlinie der Benutzerrichtlinie für den IAM-Benutzer oder die IAM-Rolle, der/die die Anforderung signiert hat.
- Vergewissern Sie sich, dass Sie den richtigen IAM-Benutzer oder die richtige IAM-Rolle verwenden. Sie können Ihren IAM-Benutzer oder Ihre IAM-Rolle überprüfen, indem Sie rechts oben in der AWS Management Console nachsehen oder den Befehl aws sts get-caller-identity verwenden.
- Prüfen Sie die IAM-Richtlinien im Zusammenhang mit dem IAM-Benutzer oder der IAM-Rolle. Sie können eine der folgenden Methoden verwenden:
  - Testen von IAM-Richtlinien mit dem IAM-Richtliniensimulator
  - Überprüfen der verschiedenen IAM-Richtlinientypen
- Bearbeiten Sie gegebenenfalls Ihre IAM-Benutzerrichtlinie.
- Sehen Sie sich die folgenden Beispiele für Richtlinien an, die den Zugriff explizit verweigern oder zulassen:
  - IAM-Benutzerrichtlinie zum expliziten Zulassen des Zugriffs: IAM: Ermöglicht und verweigert den Zugriff auf verschiedene Services, sowohl programmgesteuert als auch über die Konsole
  - Bucket-Richtlinie zum expliziten Zulassen des Zugriffs: Erteilung von Berechtigungen für mehrere Konten zum Hochladen von Objekten oder zum Festlegen von Objekt-ACLs für den öffentlichen Zugriff

- IAM-Benutzerrichtlinie zum expliziten Verweigern des Zugriffs: AWS: Verweigert den Zugriff auf AWS basierend auf der angeforderten AWS-Region
- Bucket-Richtlinie zum expliziten Verweigern des Zugriffs: SSE-KMS für alle in einen Bucket geschriebenen Objekte verlangen

Amazon-S3-ACL-Einstellungen

Wenn Sie Ihre ACL-Einstellungen überprüfen, überprüfen Sie zunächst Ihre Einstellung für Object Ownership, um festzustellen, ob ACLs für den Bucket aktiviert sind. Beachten Sie, dass ACL-Berechtigungen nur zum Erteilen von Berechtigungen und nicht zum Zurückweisen von Anforderungen verwendet werden können. ACLs können auch nicht verwendet werden, um Anforderern Zugriff zu gewähren, denen der Zugriff durch explizite Ablehnungen in Bucket-Richtlinien oder IAM-Benutzerrichtlinien verweigert wurde.

Die Einstellung für Object Ownership ist auf „Bucket-Eigentümer erzwungen“ gesetzt.

Wenn die Einstellung „Bucket-Eigentümer erzwungen“ aktiviert ist, ist es unwahrscheinlich, dass die ACL-Einstellungen einen Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) verursachen, da diese Einstellung alle ACLs deaktiviert, die für Buckets und Objekte gelten. „Bucket-Eigentümer erzwungen“ ist die Standardeinstellung (und empfohlene Einstellung) für Amazon-S3-Buckets.

Die Einstellung für Object Ownership ist auf „Bucket-Eigentümer bevorzugt“ oder „Objektschreiber“ gesetzt.

ACL-Berechtigungen sind weiterhin gültig, wenn die Einstellung „Bucket-Eigentümer bevorzugt“ oder „Objektschreiber“ verwendet wird. Es gibt zwei Arten von ACLs: Bucket-ACLs und Objekt-ACLs. Die Unterschiede zwischen diesen beiden Arten von ACLs finden Sie unter Mapping der ACL-Berechtigungen und Zugriffsrichtlinienberechtigungen.

Überprüfen Sie abhängig von der Aktion der zurückgewiesenen Anforderung die ACL-Berechtigungen für Ihren Bucket oder das Objekt:

- Wenn Amazon S3 eine `LIST`-, `PUT` (für ein Objekt), `GetBucketAcl`- oder `PutBucketAcl`-Anforderung zurückgewiesen hat, überprüfen Sie die ACL-Berechtigungen für Ihren Bucket.

> **Note**
>
> Mit den Bucket-ACL-Einstellungen können Sie keine `GET`-Objektberechtigungen gewähren.

- Wenn Amazon S3 eine `GET`-Anforderung für ein S3-Objekt oder eine PutObjectAcl-Anforderung abgelehnt hat, überprüfen Sie die ACL-Berechtigungen für das Objekt.

> **Important**
>
> Wenn es sich bei dem Konto, dem das Objekt gehört, nicht um das Konto handelt, das im Besitz des Buckets ist, wird der Zugriff auf das Objekt nicht durch die Bucket-Richtlinie gesteuert.

Beheben eines Fehlers aufgrund einer Zugriffsverweigerung (403 Forbidden) bei einer **`GET`**-Objekt-Anforderung während einer kontoübergreifenden Objekteigentümerschaft

Überprüfen Sie die Einstellungen für Object Ownership des Buckets, um den Objekteigentümer zu ermitteln. Wenn Sie Zugriff auf die Objekt-ACLs haben, können Sie auch das Konto des Objekteigentümers überprüfen. (Um das Konto des Objekteigentümers einzusehen, überprüfen Sie die Objekt-ACL-Einstellung in der Amazon-S3-Konsole.) Alternativ können Sie auch eine `GetObjectAcl`-Anforderung stellen, um die kanonische ID des Objekteigentümers zu ermitteln und so das Konto des Objekteigentümers überprüfen zu können. Standardmäßig gewähren ACLs explizite Berechtigungen für `GET`-Anforderungen an das Konto des Objekteigentümers.

Nachdem Sie bestätigt haben, dass der Objekteigentümer nicht mit dem Bucket-Eigentümer identisch ist, wählen Sie je nach Anwendungsfall und Zugriffsebene eine der folgenden Methoden aus, um den Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) zu beheben:

- ACLs deaktivieren (empfohlen) – Diese Methode gilt für alle Objekte und kann vom Bucket-Eigentümer ausgeführt werden. Bei dieser Methode erhält der Bucket-Eigentümer automatisch das Eigentum an jedem Objekt im Bucket und die volle Kontrolle darüber. Bevor Sie diese Methode implementieren, überprüfen Sie die Voraussetzungen für die Deaktivierung von ACLs. Informationen dazu, wie Sie Ihren Bucket auf den Modus „Bucket-Eigentümer erzwungen“ (empfohlen) setzen, finden Sie unter Einstellung für Object Ownership für einen vorhandenen Bucket.

> **Important**
>
> Um einen Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) zu verhindern, müssen Sie die ACL-Berechtigungen in eine Bucket-Richtlinie migrieren, bevor Sie ACLs

deaktivieren. Weitere Informationen finden Sie unter Beispiele für Bucket-Richtlinien für die Migration von ACL-Berechtigungen.

- Objekteigentümer in Bucket-Eigentümer ändern – Diese Methode kann auf einzelne Objekte angewendet werden, doch nur der Objekteigentümer (oder ein Benutzer mit den entsprechenden Berechtigungen) kann die Eigentümerschaft eines Objekts ändern. Es können zusätzliche PUT-Kosten anfallen. (Weitere Informationen finden Sie unter Amazon S3 – Preise.) Diese Methode gewährt dem Bucket-Eigentümer das vollständige Eigentum an dem Objekt, sodass der Bucket-Eigentümer den Zugriff auf das Objekt über eine Bucket-Richtlinie steuern kann.

  Führen Sie einen der folgenden Schritte aus, um die Objekteigentümerschaft zu ändern:

  - Sie (der Bucket-Eigentümer) können das Objekt wieder in den Bucket kopieren.
  - Sie können die Einstellung für Object Ownership des Buckets auf „Bucket-Eigentümer bevorzugt“ setzen. Wenn die Versionsverwaltung deaktiviert ist, werden die Objekte im Bucket überschrieben. Wenn die Versionsverwaltung aktiviert ist, werden doppelte Versionen desselben Objekts im Bucket angezeigt, für die der Bucket-Eigentümer eine Lebenszyklusregel festlegen kann, damit sie ablaufen. Anweisungen zum Ändern der Einstellung für Object Ownership finden Sie unter Einstellung für Object Ownership für einen vorhandenen Bucket.

    > **Note**
    >
    > Wenn Sie Ihre Einstellung für Object Ownership in „Bucket-Eigentümer bevorzugt“ aktualisieren, wird die Einstellung nur auf neue Objekte angewendet, die in den Bucket hochgeladen werden.

  - Sie können den Objekteigentümer das Objekt mit der vordefinierten Objekt-ACL `bucket-owner-full-control` erneut hochladen lassen.

> **Note**
>
> Für kontoübergreifende Uploads können Sie in Ihrer Bucket-Richtlinie auch die vordefinierte Objekt-ACL `bucket-owner-full-control` verlangen. Ein Beispiel für eine Bucket-Richtlinie finden Sie unter Erteilung von kontoübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt.

- Objektschreiber als Objekteigentümer beibehalten – Mit dieser Methode wird der Objekteigentümer nicht geändert, Sie können jedoch den Zugriff auf Objekte einzeln gewähren. Um Zugriff auf ein Objekt zu gewähren, müssen Sie über die Berechtigung `PutObjectAcl` für das Objekt verfügen. Um Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) zu beheben, fügen Sie dann den Anforderer als Empfänger für den Zugriff auf das Objekt in den ACLs des Objekts hinzu. Weitere Informationen finden Sie unter Konfigurieren von ACLs.

S3–Block-Public-Access-Einstellungen

Wenn die fehlgeschlagene Anforderung öffentlichen Zugriff oder öffentliche Richtlinien beinhaltet, überprüfen Sie die S3-Block-Public-Access-Einstellungen in Ihrem Konto, Bucket oder S3-Zugriffspunkt. Ab April 2023 sind alle Block-Public-Access-Einstellungen für neue Buckets standardmäßig aktiviert. Weitere Informationen dazu, wie „öffentlich“ in Amazon S3 definiert ist, finden Sie unter Die Bedeutung von „öffentlich“.

Wenn diese Option auf `TRUE` eingestellt ist, fungieren die Block-Public-Access-Einstellungen als explizite Verweigerungsrichtlinien, die die durch ACLs, Bucket-Richtlinien und IAM-Benutzerrichtlinien erteilten Berechtigungen außer Kraft setzen. Prüfen Sie die folgenden Szenarien, um festzustellen, ob Ihre Block-Public-Access-Einstellungen Ihre Anforderung zurückweisen:

- Wenn die angegebene Zugriffssteuerungsliste (ACL) öffentlich ist, weist die `BlockPublicAcls`-Einstellung Ihre `PutBucketAcl`- und `PutObjectACL`-Aufrufe zurück.
- Wenn die Anforderung eine öffentliche ACL enthält, weist die `BlockPublicAcls`-Einstellung Ihre `PutObject`-Aufrufe zurück.
- Wenn die `BlockPublicAcls`-Einstellung auf ein Konto angewendet wird und die Anforderung eine öffentliche ACL enthält, schlagen alle `CreateBucket`-Aufrufe, die öffentliche ACLs enthalten, fehl.
- Wenn die Berechtigung Ihrer Anforderung nur von einer öffentlichen ACL erteilt wird, weist die `IgnorePublicAcls`-Einstellung die Anforderung zurück.
- Wenn die angegebene Bucket-Richtlinie den öffentlichen Zugriff zulässt, weist die `BlockPublicPolicy`-Einstellung Ihre `PutBucketPolicy`-Aufrufe zurück.
- Wenn die `BlockPublicPolicy`-Einstellung auf einen Zugriffspunkt angewendet wird, schlagen alle `PutAccessPointPolicy`- und `PutBucketPolicy`-Aufrufe fehl, die eine öffentliche Richtlinie angeben und über den Zugriffspunkt erfolgen.
- Wenn der Zugriffspunkt oder Bucket über eine öffentliche Richtlinie verfügt, weist die `RestrictPublicBuckets`-Einstellung alle kontoübergreifenden Aufrufe mit Ausnahme von

AWS-Service-Prinzipalen zurück. Diese Einstellung weist auch alle anonymen (oder unsignierten) Aufrufe zurück.

Informationen zum Überprüfen und Aktualisieren Ihrer Konfigurationen der Block-Public-Access-Einstellungen finden Sie unter Konfigurieren von Block-Public-Access-Einstellungen für Ihre S3-Buckets.

Amazon-S3-Verschlüsselungseinstellungen

Amazon S3 unterstützt die serverseitige Verschlüsselung in Ihrem Bucket. Serverseitige Verschlüsselung ist die Verschlüsselung von Daten am Zielort durch die Anwendung oder den Service, der sie erhält. Amazon S3 verschlüsselt Ihre Daten auf Objektebene, wenn es diese auf Festplatten in AWS-Rechenzentren schreibt, und entschlüsselt die Daten für Sie, wenn Sie auf diese zugreifen.

Standardmäßig wendet Amazon S3 jetzt eine serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) als Basisverschlüsselung für jeden Bucket in Amazon S3 an. Amazon S3 ermöglicht es Ihnen auch, die serverseitige Verschlüsselungsmethode beim Hochladen von Objekten anzugeben.

So überprüfen Sie den Status der serverseitigen Verschlüsselung und die Verschlüsselungseinstellungen Ihres Buckets

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Bucket aus, für den Sie die Verschlüsselungseinstellungen überprüfen möchten.
4. Wählen Sie die Registerkarte Eigenschaften aus.
5. Scrollen Sie nach unten zum Abschnitt Standardverschlüsselung und sehen Sie sich die Einstellungen für den Verschlüsselungstyp an.

Verwenden Sie den Befehl get-bucket-encryption, um Ihre Verschlüsselungseinstellungen über die AWS CLI zu überprüfen.

So überprüfen Sie den Verschlüsselungsstatus eines Objekts

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
4. Wählen Sie in der Liste Objekte den Namen des Objekts aus, für das Sie eine Verschlüsselung hinzufügen oder ändern möchten.

   Die Seite mit den Objektdetails wird angezeigt.
5. Scrollen Sie nach unten zum Abschnitt Serverseitige Verschlüsselungseinstellungen, um die serverseitigen Verschlüsselungseinstellungen des Objekts anzuzeigen.

Verwenden Sie den Befehl head-object, um Ihren Objektverschlüsselungsstatus über die AWS CLI zu überprüfen.

Verschlüsselungs- und Berechtigungsanforderungen

Amazon S3 unterstützt drei Arten von serverseitiger Verschlüsselung:

- Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)
- Serverseitige Verschlüsselung mit Schlüsseln, die von AWS Key Management Service (AWS KMS) (SSE-KMS) verwaltet werden
- Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Stellen Sie unter Berücksichtigung Ihrer Verschlüsselungseinstellungen sicher, dass die folgenden Berechtigungen erfüllt sind:

- SSE-S3 – Es sind keine zusätzlichen Berechtigungen erforderlich.
- SSE-KMS (mit einem vom Kunden verwalteten Schlüssel) – Um Objekte hochzuladen, wird die Berechtigung `kms:GenerateDataKey` für AWS KMS key benötigt. Um Objekte herunterzuladen und mehrteilige Uploads von Objekten durchzuführen, ist die Berechtigung `kms:Decrypt` für den KMS-Schlüssel erforderlich.
- SSE-KMS (mit einem Von AWS verwalteter Schlüssel) – Der Anforderer muss demselben Konto angehören, das den `aws/s3`-KMS-Schlüssel besitzt. Der Anforderer muss außerdem über die richtigen Amazon-S3-Berechtigungen verfügen, um auf das Objekt zugreifen zu können.

- SSE-C (mit einem vom Kunden bereitgestellten Schlüssel) – Es sind keine zusätzlichen Berechtigungen erforderlich. Sie können die Bucket-Richtlinie so konfigurieren, dass eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln für Objekte in Ihrem Bucket erforderlich und beschränkt ist.

Wenn das Objekt mit einem vom Kunden verwalteten Schlüssel verschlüsselt ist, stellen Sie sicher, dass es Ihnen die KMS-Schlüsselrichtlinie gestattet, die Aktionen `kms:GenerateDataKey` oder `kms:Decrypt` auszuführen. Anweisungen zur Überprüfung Ihrer KMS-Schlüsselrichtlinie finden Sie unter Anzeigen einer Schlüsselrichtlinie im AWS Key Management Service-Entwicklerhandbuch.

S3-Einstellungen für die Objektsperre

Wenn in Ihrem Bucket die S3-Objektsperre aktiviert ist und das Objekt durch einen Aufbewahrungszeitraum oder eine rechtliche Aufbewahrungspflicht geschützt ist, gibt Amazon S3 beim Versuch, das Objekt zu löschen, einen Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) zurück.

So überprüfen Sie, ob für den Bucket eine Objektsperre aktiviert ist

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie überprüfen möchten.
4. Wählen Sie die Registerkarte Eigenschaften aus.
5. Scrollen Sie nach unten zum Abschnitt Objektsperre. Überprüfen Sie, ob die Einstellung für Objektsperre Aktiviert oder Deaktiviert lautet.

Um festzustellen, ob das Objekt durch einen Aufbewahrungszeitraum oder eine rechtliche Aufbewahrungspflicht geschützt ist, zeigen Sie die Sperrinformationen für Ihr Objekt an.

Wenn das Objekt durch einen Aufbewahrungszeitraum oder eine rechtliche Aufbewahrungspflicht geschützt ist, überprüfen Sie Folgendes:

- Wenn die Objektversion durch den Compliance-Aufbewahrungsmodus geschützt ist, gibt es keine Möglichkeit, sie dauerhaft zu löschen. Eine permanente `DELETE`-Anforderung eines Anforderers, einschließlich des Root-Benutzers, führt zu einem Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden). Beachten Sie außerdem, dass Amazon S3 eine Löschmarkierung für das Objekt

erstellt, wenn Sie eine `DELETE`-Anforderung für ein Objekt einreichen, das durch den Compliance-Aufbewahrungsmodus geschützt ist.

- Wenn die Objektversion durch den Governance-Aufbewahrungsmodus geschützt ist und Sie über die Berechtigung `s3:BypassGovernanceRetention` verfügen, können Sie den Schutz umgehen und die Version dauerhaft löschen. Weitere Informationen finden Sie unter Umgehen des Governance-Modus.
- Wenn die Objektversion durch eine rechtliche Aufbewahrungspflicht geschützt ist, kann eine permanente `DELETE`-Anforderung zu einem Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) führen. Um die Objektversion dauerhaft zu löschen, müssen Sie die rechtliche Aufbewahrungspflicht für die Objektversion aufheben. Um eine rechtliche Aufbewahrungspflicht aufzuheben, benötigen Sie die Berechtigung `s3:PutObjectLegalHold`. Weitere Informationen zum Aufheben einer rechtlichen Aufbewahrungspflicht finden Sie unter Konfigurieren von S3 Object Lock.

VPC-Endpunktrichtlinie

Wenn Sie über einen Virtual Private Cloud (VPC)-Endpunkt auf Amazon S3 zugreifen, stellen Sie sicher, dass die VPC-Endpunktrichtlinie Sie nicht am Zugriff auf Ihre Amazon-S3-Ressourcen hindert. Standardmäßig erlaubt die VPC-Endpunktrichtlinie alle Anforderungen an Amazon S3. Sie können die VPC-Endpunktrichtlinie auch so konfigurieren, dass bestimmte Anforderungen eingeschränkt werden. Informationen zur Überprüfung Ihrer VPC-Endpunktrichtlinie finden Sie unter Steuern des Zugriffs auf VPC-Endpoints mithilfe von Endpunktrichtlinien im AWS PrivateLink-Handbuch.

AWS Organizations-Richtlinien

Wenn Ihr AWS-Konto einer Organisation gehört, können AWS Organizations-Richtlinien Sie am Zugriff auf Amazon-S3-Ressourcen hindern. Standardmäßig blockieren AWS Organizations-Richtlinien keine Anforderungen an Amazon S3. Stellen Sie jedoch sicher, dass Ihre AWS Organizations-Richtlinien nicht so konfiguriert wurden, dass sie den Zugriff auf S3-Buckets blockieren. Anweisungen zur Überprüfung Ihrer AWS Organizations-Richtlinien finden Sie unter Auflisten aller Richtlinien im AWS Organizations-Benutzerhandbuch.

Zugriffspunkteinstellungen

Wenn Sie bei Anforderungen über Amazon-S3-Zugriffspunkte einen Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) erhalten, müssen Sie möglicherweise Folgendes überprüfen:

- Die Konfigurationen Ihrer Zugriffspunkte
- Die IAM-Benutzerrichtlinie, die für Ihre Zugriffspunkte verwendet wird
- Die Bucket-Richtlinie, die zur Verwaltung oder Konfiguration Ihrer kontoübergreifenden Zugriffspunkte verwendet wird

Zugriffspunktkonfigurationen und -richtlinien

- Wenn Sie einen Zugriffspunkt erstellen, können Sie Internet oder VPC als Netzwerkursprung festlegen. Wenn der Netzwerkursprung auf „Nur VPC“ gesetzt ist, weist Amazon S3 alle Anforderungen an den Zugriffspunkt zurück, die nicht von der angegebenen VPC stammen. Informationen zum Überprüfen des Netzwerkursprungs Ihres Zugriffspunkts finden Sie unter Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind.
- Mit Zugriffspunkten können Sie auch benutzerdefinierte Block-Public-Access-Einstellungen konfigurieren, die ähnlich wie die Block-Public-Access-Einstellungen auf Bucket- oder Kontoebene funktionieren. Informationen zu Ihren benutzerdefinierten Block-Public-Access-Einstellungen finden Sie unter Verwalten des öffentlichen Zugriffs auf Zugriffspunkte.
- Um mithilfe von Zugriffspunkten erfolgreiche Anforderungen an Amazon S3 zu stellen, stellen Sie sicher, dass der Anforderer über die erforderlichen IAM-Berechtigungen verfügt. Weitere Informationen finden Sie unter Konfigurieren von IAM-Richtlinien für die Verwendung von Zugriffspunkten.
- Wenn die Anforderung kontoübergreifende Zugriffspunkte umfasst, stellen Sie sicher, dass der Bucket-Eigentümer die Bucket-Richtlinie aktualisiert hat, um Anforderungen vom Zugriffspunkt zu autorisieren. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für kontoübergreifende Zugriffspunkte.

Wenn der Fehler aufgrund einer Zugriffsverweigerung (403 Forbidden) auch nach der Überprüfung aller Elemente in diesem Thema weiterhin besteht, rufen Sie Ihre Amazon-S3-Anforderungs-ID ab und wenden Sie sich an AWS Support, um weitere Informationen zu erhalten.

Fehlerbehebung bei Batch Operations

In den folgenden Themen werden häufige Fehler behandelt, um Sie dabei zu unterstützen, Probleme bei der Arbeit mit Batch Operations zu beheben.

Häufige Fehler

Der Auftragsbericht wird nicht bereitgestellt, wenn ein Problem mit Berechtigungen besteht oder ein Aufbewahrungsmodus der S3-Objektsperre aktiviert ist.

Der folgende Fehler tritt auf, wenn erforderliche Berechtigungen fehlen oder ein Aufbewahrungsmodus der Objektsperre (entweder Governance-Modus oder Compliance-Modus) für den Ziel-Bucket aktiviert ist.

Fehler: Fehlerursachen. Der Auftragsbericht konnte nicht in Ihren Berichts-Bucket geschrieben werden. Überprüfen Sie Ihre Berechtigungen.

Die IAM-Rolle und die Vertrauensrichtlinie müssen so konfiguriert sein, dass S3 Batch Operations Zugriff auf PUT-Objekte in dem Bucket gewährt, in dem der Bericht bereitgestellt wird. Wenn diese erforderlichen Berechtigungen fehlen, schlägt die Bereitstellung des Auftragsberichts fehl.

Wenn ein Aufbewahrungsmodus aktiviert ist, ist der Bucket mit Write-Once-Read-Many (WORM) geschützt. Die Objektsperre mit dem für den Ziel-Bucket aktivierten Aufbewahrungsmodus wird nicht unterstützt. Daher schlägt die Bereitstellung des Auftragsabschlussberichts fehl. Wählen Sie zum Beheben dieses Problems einen Ziel-Bucket für Ihre Abschlussberichte zu Aufträgen aus, für den kein Aufbewahrungsmodus der Objektsperre aktiviert ist.

S3-Batchreplikation mit folgendem Fehler fehlgeschlagen: Bei der Manifestgenerierung wurden keine Schlüssel gefunden, die den Filterkriterien entsprechen

Der folgende Fehler tritt auf, wenn Objekte im Quell-Bucket in der Speicherklasse S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive gespeichert sind.

Fehler: Bei der Manifestgenerierung wurden keine Schlüssel gefunden, die den Filterkriterien entsprechen.

> **Note**
>
> Wenn die angegebenen Filterkriterien mit keinen gültigen Objekten im Quell-Bucket übereinstimmen, entspricht dieser Fehler möglicherweise dem erwarteten Verhalten. Überprüfen Sie die Filterkriterien und die Speicherklasse des Zielobjekts.

Um die Batchreplikation für diese Objekte zu verwenden, stellen Sie sie zunächst in der Speicherklasse S3 Standard wieder her, indem Sie die Operation S3 Initiate Restore Object in einem Batch-Operations-Auftrag verwenden. Weitere Informationen finden Sie unter Wiederherstellen eines archivierten Objekts und Wiederherstellen von Objekten (Batch Operations). Nachdem Sie die Objekte wiederhergestellt haben, können Sie sie mithilfe eines Batchreplikationsauftrags replizieren.

Fehler bei Batch Operations nach dem Hinzufügen einer neuen Replikationsregel zu einer vorhandenen Replikationskonfiguration

Batch Operations versucht, eine vorhandene Objektreplikation für jede Regel in der Replikationskonfiguration des Quell-Buckets auszuführen. Wenn es Probleme mit einer der vorhandenen Replikationsregeln gibt, können Fehler auftreten.

Im Abschlussbericht des Batch-Operations-Auftrags werden die Ursachen für das Fehlschlagen des Auftrags erläutert. Eine Liste mit häufig auftretenden Fehlern finden Sie unter Gründe für das Fehlschlagen der Replikation in Amazon S3.

Fehlschlagen von Objekten in Batch Operations mit dem Fehler 400 InvalidRequest: Auftrag aufgrund fehlender Versions-ID fehlgeschlagen

Der folgende Beispielfehler tritt auf, wenn ein Batch-Operations-Auftrag Aktionen für Objekte in einem versionsfähigen Bucket ausführt und im Manifest ein Objekt mit leerem Versions-ID-Feld feststellt.

Fehler: *`BUCKETNAME,Präfix/Dateiname`*,fehlgeschlagen,400,InvalidRequest,Auftrag aufgrund fehlender Versions-ID fehlgeschlagen

Dieser Fehler tritt auf, weil das Versions-ID-Feld im Manifest eine leere Zeichenfolge und nicht die Literalzeichenfolge `null` ist.

Batch Operations schlägt für diese(s) spezielle(n) Objekt(e) fehl, aber nicht für den gesamten Auftrag. Dieses Problem tritt auf, wenn das Manifestformat so konfiguriert ist, dass während der Operation Versions-IDs verwendet werden. Bei nicht versionfähigen Aufträgen tritt dieses Problem nicht auf, da sie nur mit der aktuellen Version der einzelnen Objekte arbeiten und die Versions-IDs im Manifest ignorieren.

Wandeln Sie zum Beheben dieses Problems die leeren Versions-IDs in die Zeichenfolge `null` um. Weitere Informationen finden Sie unter the section called “Konvertieren leerer Versions-ID-Zeichenfolgen in Null-Zeichenfolgen”.

Fehler bei Auftragserstellung mit aktivierter Auftrags-Tag-Option

Ohne die Berechtigung `s3:PutJobTagging` führt das Erstellen von Batch-Operations-Aufträgen mit aktivierter Auftrags-Tag-Option zum Fehler `403 access denied`.

Um Batch-Operations-Aufträge mit aktivierter Auftrags-Tag-Option erstellen zu können, muss der AWS Identity and Access Management (IAM)-Benutzer, der den Batch-Operations-Auftrag erstellt, zusätzlich zur Berechtigung `s3:CreateJob` über die Berechtigung `s3:PutJobTagging` verfügen.

Weitere Informationen zu den erforderlichen Berechtigungen für Batch Operations finden Sie unter the section called “Gewähren von Berechtigungen”.

Zugriff zum Lesen des Manifests verweigert

Wenn Sie versuchen, einen Batch-Operations-Auftrag zu erstellen und Batch Operations die Manifestdatei nicht lesen kann, können die folgenden Fehler auftreten.

AWS CLI

Fehlerursache Lesen des Manifests verboten: AccessDenied

Amazon S3-Konsole

Warnung: Das ETag des Manifestobjekts kann nicht abgerufen werden. Geben Sie ein anderes Objekt an, um fortzufahren.

Führen Sie die folgenden Schritte aus, um dieses Problem zu beheben:

- Stellen Sie sicher, dass die IAM-Rolle für das AWS-Konto, das Sie zum Erstellen des Batch-Operations-Auftrags verwendet haben, über `s3:GetObject`-Berechtigungen verfügt. Die IAM-Rolle des Kontos muss über `s3:GetObject`-Berechtigungen verfügen, damit Batch Operations die Manifestdatei lesen kann.

  Weitere Informationen zu den erforderlichen Berechtigungen für Batch Operations finden Sie unter the section called “Gewähren von Berechtigungen”.
- Überprüfen Sie die Metadaten der Manifestobjekte auf etwaige Zugriffskonflikte mit S3 Object Ownership. Informationen zu S3 Object Ownership finden Sie unter the section called “Steuern der Objekteigentümerschaft”.
- Prüfen Sie, ob AWS Key Management Service (AWS KMS)-Schlüssel zum Verschlüsseln der Manifestdatei verwendet werden.

  Batch Operations unterstützen CSV-Bestandsberichte, die mit AWS KMS verschlüsselt sind. Batch Operations unterstützen jedoch keine CSV-Manifestdateien, die mit AWS KMS verschlüsselt sind. Weitere Informationen finden Sie unter Konfigurieren von Amazon S3 Inventory und Angeben eines Manifests.

CORS-Fehlerbehebung

Wenn Sie ein unerwartetes Verhalten feststellen, wenn Sie auf Buckets zugreifen, die mit CORS-Konfiguration eingerichtet wurden, führen Sie die folgenden Schritte zur Fehlerbehebung durch:

1. Vergewissern Sie sich, dass die CORS-Konfiguration für den Bucket eingerichtet ist.

   Falls die CORS-Konfiguration eingerichtet wurde, zeigt die Konsole den Link Edit CORS Configuration (CORS-Konfiguration bearbeiten) im Bereich Permissions (Berechtigungen) des Buckets Properties (Eigenschaften) an.

2. Erfassen Sie die vollständige Anfrage und antworten Sie mit einem Tool Ihrer Wahl. Es für jede von Amazon S3 erhaltene Anfrage eine CORS-Regel geben, die mit den Daten in Ihrer Anfrage wie folgt übereinstimmt:

   a. Stellen Sie sicher, dass die Anfrage den Origin-Header besitzt.

      Falls der Header fehlt, verarbeitet Amazon S3 die Anfrage nicht als ursprungsübergreifende Anfrage und sendet in der Antwort keine CORS-Antwort-Header.

   b. Stellen Sie sicher, dass der Origin-Header in Ihrer Anfrage mit mindestens einem der `AllowedOrigin`-Elemente in der angegebenen `CORSRule` übereinstimmt.

      Das Schema, der Host und die Port-Werte im Origin-Anfrageheader müssen mit den `AllowedOrigin`-Elementen in der `CORSRule` übereinstimmen. Wenn Sie beispielsweise die `CORSRule` so eingerichtet haben, dass der Ursprung `http://www.example.com` zulässig ist, stimmen die Ursprünge `https://www.example.com` und `http://www.example.com:80` in Ihrer Anfrage nicht mit dem in Ihrer Konfiguration erlaubten Ursprung überein.

   c. Stellen Sie sicher, dass die Methode in Ihrer Anfrage (oder in einer Preflight-Anfrage die in `Access-Control-Request-Method` angegebene Methode) eines der `AllowedMethod`-Elemente in derselben `CORSRule` ist.

   d. Wenn bei einer Preflight-Anfrage die anfrage einen `Access-Control-Request-Headers`-Header enthält, überprüfen Sie, ob die `CORSRule` die `AllowedHeader`-Einträge für jeden Wert im `Access-Control-Request-Headers`-Header enthält.

Fehlerbehebung bei Problemen mit dem Amazon-S3-Lebenszyklus

Die folgenden Informationen können Ihnen helfen, häufiger auftretende Probleme mit den Lebenszyklusregeln von Amazon S3 zu beheben.

Themen

- Wie leere ich meinen S3-Bucket mithilfe von Lebenszyklusregeln?
- Meine Abrechnung für Amazon S3 weist nach der Übergabe von Objekten in eine kostengünstigere Speicherklasse höhere Kosten auf.
- Ich habe meine Bucket-Richtlinie aktualisiert, meine S3-Objekte werden jedoch noch immer aufgrund abgelaufener Lebenszyklusregeln gelöscht.
- Kann ich S3-Objekte wiederherstellen, die aufgrund von S3-Lebenszyklusregeln abgelaufen sind?

Ich habe eine Auflistungsoperation für meinen Bucket ausgeführt und es wurden Objekte angezeigt, von denen ich dachte, dass sie abgelaufen oder aufgrund einer Lebenszyklusregel übergeben worden waren.

Das Übergeben von Objekten und das Ablaufen von Objekten in S3-Lebenszyklen sind asynchrone Operationen. Daher kann es zu einer Verzögerung zwischen dem Zeitpunkt kommen, zu dem die Objekte für ein Ablaufen oder eine Übergabe infrage kommen, und dem Zeitpunkt, zu dem sie tatsächlich übergeben werden oder ablaufen. Änderungen an der Abrechnung werden sofort nach Erfüllung der Lebenszyklusregel angewendet, auch wenn die Aktion noch nicht abgeschlossen ist. Eine Ausnahme von diesem Verhalten tritt auf, wenn Sie eine Lebenszyklusregel für die Übergabe in die Speicherklasse S3 Intelligent Tiering festgelegt haben. In diesem Fall treten Abrechnungsänderungen erst auf, wenn das Objekt in S3 Intelligent-Tiering übergeben worden ist. Weitere Informationen zu Änderungen der Abrechnung finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

> **Note**
>
> Amazon S3 übergibt keine Objekte, die kleiner als 128 KB sind, von der Speicherklasse S3 Standard oder S3 Standard-IA in die Klassen S3 Intelligent-Tiering, S3 Standard-IA oder S3 One Zone-IA.

Wie überwache ich den Fortschritt meiner Lebenszyklusregel, um sicherzustellen, dass sie aktiv ist?

Verwenden Sie das Storage-Lens-Dashboard, um den Fortschritt aller aktiven Lebenszyklusregeln anzuzeigen (oder Änderungen zu überwachen). Über das Dashboard können Sie die folgenden Metriken anzeigen, die die Anzahl oder Größe der Objekte überwachen.

- Aktuelle Versions-Bytes
- Anzahl der Objekte der aktuellen Version
- Nicht aktuelle Versions-Bytes
- Anzahl nicht aktueller Versionsobjekte
- Anzahl der Löschmarkierungsobjekte
- Markierungsspeicherbytes löschen
- Unvollständige Bytes für mehrteilige Uploads
- Anzahl unvollständiger mehrteiliger Uploads

Sie können auch die folgenden Funktionen zum Überwachen Ihrer Lebenszyklusregeln verwenden:

- Amazon S3 Inventory – Sie können S3 Inventory verwenden, um die Liste der Präfixe oder Objekte für den Amazon-S3-Bucket (in CSV), in Apache Optimized Row Columnar (ORC) oder im Format Apache Parquet für Prüfungszwecke zu generieren. Abhängig von Ihrem Anwendungsfall können Sie S3 Inventory auch in Standard-SQL mit Amazon Athena abfragen.
- S3-Ereignisbenachrichtigungen – Sie können Ereignisbenachrichtigungen einrichten, damit Sie über alle Lebenszyklusereignisse im Zusammenhang mit dem Ablaufen oder mit Übergaben informiert werden.
- S3-Serverzugriffsprotokolle – Sie können Serverzugriffsprotokolle für den S3-Bucket aktivieren, um Aktionen im Zusammenhang mit dem Lebenszyklus zu erfassen, wie beispielsweise Objektübergaben in eine andere Speicherklasse und das Ablaufen von Objekten. Weitere Informationen finden Sie unter Lebenszyklus und Protokollieren.

Die Anzahl meiner S3-Objekte steigt weiterhin an, obwohl ich Lebenszyklusregeln für einen Bucket mit aktivierter Versionsverwaltung eingerichtet habe.

Wenn ein Objekt in einem Bucket mit aktivierter Versionsverwaltung als abgelaufen festgelegt wird, wird das Objekt nicht vollständig aus dem Bucket gelöscht. Stattdessen wird eine Löschmarkierung als neueste Version des Objekts erstellt. Löschmarkierungen werden weiterhin als Objekte gezählt. Wenn also die Lebenszyklusregel erstellt wird, dass nur die aktuellen Versionen ablaufen, erhöht sich die Anzahl der Objekte im S3-Bucket, anstatt zu sinken.

Nehmen wir beispielsweise an, dass für einen S3-Bucket mit 100 Objekten die Versionsverwaltung aktiviert und die Lebenszyklusregel festgelegt ist, dass aktuelle Versionen des Objekts nach 7 Tagen ablaufen. Nach dem siebten Tag erhöht sich die Anzahl der Objekte auf 200, da zusätzlich zu den 100 ursprünglichen Objekten, bei denen es sich jetzt um die nicht aktuellen Versionen handelt, 100 Löschmarkierungen erstellt werden. Weitere Informationen zu den Aktionen der S3-Lebenszyklus-Konfigurationsregel für Buckets mit aktivierter Versionsverwaltung finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

Um Objekte dauerhaft zu entfernen, fügen Sie eine zusätzliche Lebenszykluskonfiguration hinzu, um die vorherigen Versionen der Objekte, abgelaufene Löschmarkierungen und unvollständige mehrteilige Uploads zu löschen. Anweisungen zum Erstellen neuer Lebenszyklusregeln finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

> **Note**
>
> - Amazon S3 rundet das Übergabe- oder Ablaufdatum eines Objekts auf Mitternacht UTC am nächsten Tag auf. Weitere Informationen finden Sie unter Lebenszyklusregeln: Basierend auf dem Alter eines Objekts.
> - Bei S3-Objekten, die durch die Objektsperre geschützt sind, werden aktuelle Versionen nicht dauerhaft gelöscht. Stattdessen wird den Objekten eine Löschmarkierung hinzugefügt, sodass sie nicht mehr aktuell sind. Nicht aktuelle Versionen werden dann beibehalten und laufen nicht dauerhaft ab.

Wie leere ich meinen S3-Bucket mithilfe von Lebenszyklusregeln?

S3-Lebenszyklusregeln sind ein effektives Tool zum Leeren eines S3-Buckets mit Millionen von Objekten. Wenn Sie eine große Anzahl von Objekten aus Ihrem S3-Bucket löschen möchten, stellen Sie sicher, dass Sie diese vier Paare von Lebenszyklusregeln verwenden:

- Aktuelle Objektversionen ablaufen lassen und Vorherige Versionen von Objekten dauerhaft löschen
- Löschmarkierungen für abgelaufenes Objekt löschen und Unvollständige mehrteilige Uploads löschen

Anweisungen zum Erstellen einer neuen Lebenszyklus-Konfigurationsregel finden Sie unter Festlegen der Lebenszyklus-Konfiguration für einen Bucket.

> **Note**
>
> Bei S3-Objekten, die durch die Objektsperre geschützt sind, werden aktuelle Versionen nicht dauerhaft gelöscht. Stattdessen wird den Objekten eine Löschmarkierung hinzugefügt, sodass sie nicht mehr aktuell sind. Nicht aktuelle Versionen werden dann beibehalten und laufen nicht dauerhaft ab.

Meine Abrechnung für Amazon S3 weist nach der Übergabe von Objekten in eine kostengünstigere Speicherklasse höhere Kosten auf.

Es gibt mehrere Gründe, warum Ihre Abrechnung nach der Übergabe von Objekten in eine kostengünstigere Speicherklasse höhere Kosten aufweisen kann:

- S3-Glacier-Overheadgebühren für kleine Objekte

  Bei jedem Objekt, das an S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive übergeben wird, ist ein Gesamt-Overhead von 40 KB mit dieser Speicheraktualisierung verbunden. Im Rahmen des Overheads von 40 KB werden 8 KB zum Speichern der Metadaten und des Objektnamens verwendet. Diese 8 KB werden gemäß den S3-Standardgebühren berechnet. Die restlichen 32 KB werden für die Indexierung und die zugehörigen Metadaten verwendet. Diese 32 KB werden gemäß den Gebühren von S3 Glacier Flexible Retrieval oder S3 Glacier Deep Archive berechnet.

  Wenn Sie viele kleinere Objekte speichern, sollten Sie daher keine Lebenszyklusübergaben verwenden. Stattdessen sollten Sie viele kleinere Objekte zu einer kleineren Anzahl von großen Objekten zusammenzufassen, bevor Sie sie in Amazon S3 speichern, um so die Overheadgebühren zu reduzieren. Weitere Informationen zu den Kostenüberlegungen finden Sie unter Übergang in die Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive (Objektarchivierung).
- Mindestspeichergebühren

  Für einige S3-Speicherklassen gelten Mindestanforderungen an die Speicherdauer. Für Objekte, die vor Ablauf der Mindestdauer gelöscht, überschrieben oder aus diesen Klassen übergeben werden, wird eine anteilige Gebühr für die vorzeitige Übergabe oder das Löschen berechnet. Diese Mindestanforderungen an die Speicherdauer lauten wie folgt:

  - S3 Standard-IA und S3 One Zone-IA – 30 Tage

- S3 Glacier Flexible Retrieval und S3 Glacier Instant Retrieval – 90 Tage
- S3 Glacier Deep Archive – 180 Tage

Weitere Informationen zu diesen Anforderungen finden Sie im Abschnitt Beschränkungen unter Übergang von Objekten mit S3-Lebenszyklus. Allgemeine S3-Preisinformationen finden Sie unter Amazon S3 – Preise und im AWS Pricing Calculator.

- Kosten für die Lebenszyklusübergabe

  Immer wenn ein Objekt durch eine Lebenszyklusregel in eine andere Speicherklasse übergeben wird, zählt Amazon S3 diese Übergabe als eine Übergabeanforderung. Die Kosten für diese Übergabeanforderungen kommen zu den Kosten dieser Speicherklassen hinzu. Wenn Sie vorhaben, sehr viele Objekte zu übergeben, sollten Sie die Anforderungskosten für Übergaben in niedrigere Stufen berücksichtigen. Weitere Informationen finden Sie unter Amazon S3 – Preise.

Ich habe meine Bucket-Richtlinie aktualisiert, meine S3-Objekte werden jedoch noch immer aufgrund abgelaufener Lebenszyklusregeln gelöscht.

`Deny`-Anweisungen in einer Bucket-Richtlinie verhindern den Ablauf der in einer Lebenszyklusregel definierten Objekte nicht. Lebenszyklusaktionen (wie Übergaben oder der Ablauf) verwenden die S3-Operation `DeleteObject` nicht. Stattdessen werden S3-Lebenszyklusaktionen mithilfe interner S3-Endpunkte ausgeführt. (Weitere Informationen finden Sie unter Lebenszyklus und Protokollieren.)

Um zu verhindern, dass Ihre Lebenszyklusregel Aktionen ausführt, müssen Sie die Regel bearbeiten, löschen oder deaktivieren.

Kann ich S3-Objekte wiederherstellen, die aufgrund von S3-Lebenszyklusregeln abgelaufen sind?

Die einzige Möglichkeit, aufgrund des S3-Lebenszyklus abgelaufene Objekte wiederherzustellen, besteht in der Versionsverwaltung. Diese muss vor dem Ablauf der Objekte aktiviert sein. Sie können die Ablaufoperationen, die durch Lebenszyklusregeln ausgeführt werden, nicht rückgängig machen. Wenn Objekte aufgrund der geltenden S3-Lebenszyklusregeln dauerhaft gelöscht werden, können Sie diese Objekte nicht wiederherstellen. Informationen zum Aktivieren der Versionsverwaltung für einen Bucket finden Sie unter the section called “Verwenden der S3-Versioning”.

Wenn Sie die Versionsverwaltung auf den Bucket angewendet haben und die nicht aktuellen Versionen der Objekte noch intakt sind, können Sie frühere Versionen der abgelaufenen Objekte

wiederherstellen. Weitere Informationen zum Verhalten der Aktionen von S3-Lebenszyklusregeln und zu Versionsverwaltungsstatus finden Sie in der Tabelle Lebenszyklus-Aktionen und der Versioning-Status eines Buckets in Elemente, die Lebenszyklus-Aktionen beschreiben.

> **Note**
>
> Wenn der S3-Bucket durch AWS-Backups oder S3 Replication geschützt ist, können Sie diese Funktionen möglicherweise auch verwenden, um Ihre abgelaufenen Objekte wiederherzustellen.

Fehlerbehebung bei einer Replikation

In diesem Abschnitt finden Sie Tipps zur Problembehandlung in Amazon S3 Replication und Informationen zu Fehlern in S3 Batch Replication.

Themen

- Tipps zur Fehlerbehebung in S3 Replication
- Fehler bei der Batchreplikation

Tipps zur Fehlerbehebung in S3 Replication

Wenn Objektreplikate nicht im Ziel-Bucket erscheinen, nachdem Sie die Replikation konfiguriert haben, können Sie mit diesen Tipps zur Fehlerbehebung Probleme identifizieren und beheben.

- Die meisten Objekte werden innerhalb von 15 Minuten repliziert. Die von Amazon S3 für die Replikation eines Objekts benötigte Zeit hängt von verschiedenen Faktoren ab, einschließlich des Quell- und Ziel-Regionspaars sowie der Größe des Objekts. Bei großen Objekten kann die Replikation mehrere Stunden dauern. Sie können Begrenzung der S3-Replikationszeit (S3 RTC) verwenden, um einen Überblick über die Replikationszeiten zu erhalten.

  Wenn das replizierte Objekt groß ist, warten Sie eine Weile, bevor Sie überprüfen, ob es im Ziel erscheint. Sie können auch den Replikationsstatus des Quellobjekts überprüfen. Wenn der Replikationsstatus des Objekts `PENDING` lautet, hat Amazon S3 die Replikation noch nicht abgeschlossen. Wenn der Replikationsstatus des Objekts `FAILED` lautet, überprüfen Sie die Replikationskonfiguration des Quell-Buckets. Wenn Sie Informationen zu Fehlern während der Replikation erhalten möchten, können Sie einrichten, dass Sie über Fehlerereignisse bei

der Replikation mit Amazon-S3-Ereignisbenachrichtigungen benachrichtigt werden. Weitere Informationen finden Sie unter Erhalten von Amazon-S3-Ereignisbenachrichtigungen über Replikations-Fehlerereignisse.

- Sie können die API-Operation `HeadObject` aufrufen, um den Replikationsstatus eines Objekts zu überprüfen. Die API-Operation `HeadObject` gibt den Replikationsstatus `PENDING`, `COMPLETED` oder `FAILED` eines Objekts zurück. Als Antwort auf den API-Aufruf `HeadObject` wird der Replikationsstatus im Element `x-amz-replication-status` zurückgegeben.

  > **Note**
  >
  > Zum Ausführen von `HeadObject` müssen Sie über Lesezugriff auf das angeforderte Objekt verfügen. Eine `HEAD`-Anforderung bietet die gleichen Optionen wie eine `GET`-Anforderung, ohne eine `GET`-Operation auszuführen. Um beispielsweise eine `HeadObject`-Anforderung mit der AWS Command Line Interface (AWS CLI) auszuführen, können Sie den folgenden Befehl ausführen. Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.
  >
  > ```
  > aws s3api head-object --bucket my-bucket --key index.html
  > ```

- Nachdem `HeadObject` die Objekte mit dem Replikationsstatus `FAILED` zurückgegeben hat, können Sie S3 Batch Replication verwenden, um diese fehlgeschlagenen Objekte zu replizieren. Alternativ können Sie die fehlgeschlagenen Objekte erneut in den Quell-Bucket hochladen, wodurch die Replikation für die neuen Objekte initiiert wird.
- Überprüfen Sie in der Replikations-Konfiguration des Quell-Buckets Folgendes:
  - ob der Amazon-Ressourcennamen (ARN) des Ziel-Buckets korrekt ist.
  - ob das Schlüsselnamenpräfix korrekt ist. Wenn Sie beispielsweise die Konfiguration so einrichten, dass nur Objekte mit dem Präfix `Tax` repliziert werden, werden nur Objekte mit Schlüsselnamen wie beispielsweise `Tax/document1` oder `Tax/document2` repliziert. Ein Objekt mit dem Schlüsselnamen `document3` wird nicht repliziert.
  - Der Status der Replikationsregel lautet `Enabled`.
- Stellen Sie sicher, dass die Versionsverwaltung in der Replikationskonfiguration der Buckets nicht ausgesetzt wurde. Sowohl für die Quell- als auch für die Ziel-Buckets muss die Versionsverwaltung aktiviert sein.
- Wenn eine Replikationsregel auf Objekteigentümerschaft in Eigentümer des Ziel-Buckets ändern gesetzt ist, muss die für die Replikation verwendete AWS Identity and Access Management

(IAM)-Rolle über die Berechtigung `s3:ObjectOwnerOverrideToBucketOwner` verfügen. Diese Berechtigung wird für die Ressource (in diesem Fall den Ziel-Bucket) erteilt. Die folgende `Resource`-Anweisung zeigt beispielsweise, wie diese Berechtigung für den Ziel-Bucket erteilt wird:

```
{
  "Effect":"Allow",
  "Action":[
    "s3:ObjectOwnerOverrideToBucketOwner"
  ],
  "Resource":"arn:aws:s3:::DestinationBucket/*"
}
```

- Wenn sich der Ziel-Bucket im Besitz eines anderen Kontos befindet, muss der Eigentümer des Ziel-Buckets die Berechtigung `s3:ObjectOwnerOverrideToBucketOwner` über die Ziel-Bucket-Richtlinie auch dem Eigentümer des Quell-Buckets erteilen. Wenn Sie die folgende Bucket-Beispielrichtlinie verwenden möchten, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen:

```
{
  "Version": "2012-10-17",
  "Id": "Policy1644945280205",
  "Statement": [
    {
      "Sid": "Stmt1644945277847",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789101:role/s3-replication-role"
      },
      "Action": [
        "s3:ReplicateObject",
        "s3:ReplicateTags",
        "s3:ObjectOwnerOverrideToBucketOwner"
      ],
      "Resource": "arn:aws:s3:::DestinationBucket/*"
    }
  ]
}
```

> **Note**
>
> Wenn die Einstellungen des Ziel-Buckets zur Objekteigentümerschaft Bucket-Eigentümer erzwungen beinhalten, müssen Sie die Einstellung in der Replikationsregel nicht in Objekteigentümerschaft in Eigentümer des Ziel-Buckets ändern ändern. Die Änderung der Objekteigentümerschaft erfolgt standardmäßig. Weitere Informationen zum Ändern der Replikateigentümerschaft finden Sie unter Ändern des Replikat-Eigentümers.

- Wenn Sie die Replikationskonfiguration in einem kontoübergreifenden Szenario festlegen, in dem die Quell- und Ziel-Buckets verschiedenen gehörenAWS-Konten, können die Ziel-Buckets nicht als Buckets mit Zahlung durch den Anforderer konfiguriert werden. Weitere Informationen finden Sie unter Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragungen und Nutzung.
- Wenn die Quellobjekte eines Buckets mit einem AWS Key Management Service (AWS KMS)-Schlüssel verschlüsselt sind, muss die Replikationsregel so konfiguriert werden, dass sie mit AWS KMS verschlüsselte Objekte umfasst. Wählen Sie Mit AWS KMS verschlüsselte Objekte replizieren unter den Einstellungen zu Verschlüsselung in der Amazon-S3-Konsole aus. Wählen Sie dann einen AWS KMS-Schlüssel zum Verschlüsseln der Zielobjekte aus.

> **Note**
>
> Wenn sich der Ziel-Bucket in einem anderen Konto befindet, geben Sie einen vom Kunden verwalteten AWS KMS-Schlüssel an, der dem Zielkonto gehört. Verwenden Sie nicht den von Amazon S3 verwalteten Standardschlüssel (`aws/s3`). Bei Verwendung des Standardschlüssels werden die Objekte mit dem von Amazon S3 verwalteten Schlüssel verschlüsselt, der dem Quellkonto gehört. Dadurch wird verhindert, dass das Objekt für ein anderes Konto freigegeben wird. Infolgedessen kann das Zielkonto nicht auf die Objekte im Ziel-Bucket zugreifen.

Um zum Verschlüsseln der Zielobjekte einen AWS KMS-Schlüssel zu verwenden, der zum Zielkonto gehört, muss das Zielkonto der Replikationsrolle in der KMS-Schlüsselrichtlinie die Berechtigungen `kms:GenerateDataKey` und `kms:Encrypt` erteilen. Wenn Sie die folgende Beispielanweisung in Ihrer KMS-Schlüsselrichtlinie verwenden möchten, ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen:

```
{
    "Sid": "AllowS3ReplicationSourceRoleToUseTheKey",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::123456789101:role/s3-replication-role"
    },
    "Action": ["kms:GenerateDataKey", "kms:Encrypt"],
    "Resource": "*"
}
```

Wenn Sie ein Sternchen (*) für die `Resource`-Anweisung in der AWS KMS-Schlüsselrichtlinie verwenden, erteilt die Richtlinie nur der Replikationsrolle die Berechtigung, den KMS-Schlüssel zu verwenden. Die Richtlinie erlaubt der Replikationsrolle nicht, ihre Berechtigungen zu erweitern.

Standardmäßig gewährt die KMS-Schlüsselrichtlinie dem Root-Benutzer volle Berechtigungen für den Schlüssel. Diese Berechtigungen können an andere Benutzer in demselben Konto delegiert werden. Sofern die Quell-KMS-Schlüsselrichtlinie keine `Deny`-Anweisungen enthält, reicht es aus, eine IAM-Richtlinie zu verwenden, um der Replikationsrolle Berechtigungen für den Quell-KMS-Schlüssel zu erteilen.

> **Note**
>
> KMS-Schlüsselrichtlinien, die den Zugriff auf bestimmte CIDR-Bereiche, VPC-Endpunkte oder S3-Zugriffspunkte einschränken, können dazu führen, dass die Replikation fehlschlägt.

Wenn je nach Verschlüsselungskontext entweder der Quell- oder der Ziel-KMS-Schlüssel Berechtigungen erteilt, vergewissern Sie sich, dass Amazon-S3-Bucket-Schlüssel für die Buckets aktiviert sind. Wenn die S3-Bucket-Schlüssel für die Buckets aktiviert sind, muss der Verschlüsselungskontext die Ressource auf Bucket-Ebene sein, wie im Folgenden veranschaulicht:

```
"kms:EncryptionContext:arn:aws:arn": [
     "arn:aws:s3:::SOURCE_BUCKET_NAME"
     ]
"kms:EncryptionContext:arn:aws:arn": [
     "arn:aws:s3:::DESTINATION_BUCKET_NAME"
     ]
```

Zusätzlich zu den durch die KMS-Schlüsselrichtlinie gewährten Berechtigungen muss das Quellkonto der IAM-Richtlinie der Replikationsrolle die folgenden Mindestberechtigungen hinzufügen:

```
{
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": [
        "SourceKmsKeyArn"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "kms:GenerateDataKey",
        "kms:Encrypt"
    ],
    "Resource": [
        "DestinationKmsKeyArn"
    ]
}
```

Weitere Informationen zur Replikation von mit AWS KMS verschlüsselten Objekten finden Sie unter Replizieren verschlüsselter Objekte.

- Wenn sich der Ziel-Bucket im Besitz eines anderen AWS-Konto befindet, stellen Sie sicher, dass der Bucket-Eigentümer eine Bucket-Richtlinie für den Ziel-Bucket eingerichtet hat, die dem Eigentümer des Quell-Buckets die Replikation von Objekten gestattet. Ein Beispiel finden Sie unter Konfigurieren der Replikation, wenn sich Quell- und Ziel-Buckets im Eigentum verschiedener Konten befinden.
- Wenn Ihre Objekte auch nach der Bestätigung der Berechtigungen nicht repliziert werden, suchen Sie an den folgenden Stellen nach expliziten `Deny`-Anweisungen:
  - `Deny`-Anweisungen in den Richtlinien des Quell- oder Ziel-Buckets. Die Replikation schlägt fehl, wenn die Bucket-Richtlinie den Zugriff auf die Replikationsrolle für eine der folgenden Aktionen verweigert:

    Quell-Bucket

```
        "s3:GetReplicationConfiguration",
        "s3:ListBucket",
        "s3:GetObjectVersionForReplication",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
```

Ziel-Buckets:

```
        "s3:ReplicateObject",
        "s3:ReplicateDelete",
        "s3:ReplicateTags"
```

  - `Deny`-Anweisungen oder Berechtigungsgrenzen, die der IAM-Rolle zugeordnet sind, können dazu führen, dass die Replikation fehlschlägt.
  - `Deny`-Anweisungen in den Service-Kontrollrichtlinien von AWS Organizations, die entweder dem Quell- oder dem Zielkonto zugeordnet sind, können dazu führen, dass die Replikation fehlschlägt.
- Wenn ein Objektreplikat nicht im Ziel-Bucket angezeigt wird, könnten folgende Probleme die Replikation verhindert haben:
  - Amazon S3 repliziert keine Objekte in einem Quell-Bucket, der ein Replikat ist, das mit einer anderen Replikations-Konfiguration erstellt wurde. Wenn Sie beispielsweise die Replikationskonfiguration von Bucket A zu Bucket B zu Bucket C festlegen, repliziert Amazon S3 keine Objektreplikate von Bucket B zu Bucket C.
  - Ein Quell-Bucket-Eigentümer kann anderen AWS-Konten Berechtigungen für das Hochladen von Objekten erteilen. Standardmäßig besitzt der Quell-Bucket-Eigentümer keine Berechtigungen für die Objekte, die von anderen Konten erstellt wurden. Die Replikations-Konfiguration repliziert nur die Objekte, für die der Quell-Bucket-Eigentümer über Zugriffsberechtigungen verfügt. Der Bucket-Eigentümer kann anderen AWS-Konten Berechtigungen zum bedingten Erstellen von Objekten erteilen. Dabei sind explizite Zugriffsberechtigungen für diese Objekte erforderlich. Eine Beispielrichtlinie finden Sie unter Erteilung von kontoübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt.

- Angenommen, Sie fügen einer Replikationskonfiguration eine Regel hinzu, um eine Teilmenge von Objekten mit einem spezifischen Tag zu replizieren. In diesem Fall müssen Sie den spezifischen Tag-Schlüssel und -Wert zum Zeitpunkt der Objekterstellung zuweisen, damit Amazon S3 das Objekt replizieren kann. Wenn Sie zuerst ein Objekt erstellen und dann dem vorhandenen Objekt das Tag hinzufügen, repliziert Amazon S3 das Objekt nicht.
- Verwenden Sie Amazon-S3-Ereignisbenachrichtigungen, um sich informieren zu lassen, wenn Objekte nicht in ihre Ziel-AWS-Region repliziert wurden. Amazon-S3-Ereignisse sind über Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS) oder AWS Lambda verfügbar. Weitere Informationen finden Sie unter Erhalten von Amazon-S3-Ereignisbenachrichtigungen über Replikations-Fehlerereignisse.

  Sie können Amazon-S3-Ereignisbenachrichtigungen auch verwenden, um die Ursachen für fehlgeschlagene Replikationen anzuzeigen. Eine Liste der Ursachen für fehlgeschlagene Replikationen finden Sie unter Gründe für das Fehlschlagen der Replikation in Amazon S3.

Fehler bei der Batchreplikation

Um Fehler bei Objekten zu beheben, die nicht in den Ziel-Bucket repliziert werden, überprüfen Sie die verschiedenen Arten von Berechtigungen für den Bucket, die Replikationsrolle und die IAM-Rolle, die zum Erstellen des Batchreplikationsauftrags verwendet werden. Überprüfen Sie außerdem die Einstellungen für den öffentlichen Zugriff und die Einstellungen für die Bucket-Eigentümerschaft.

Während der Batchreplikation können die folgenden Fehler auftreten:

- Die Batchoperation ist aus folgendem Grund fehlgeschlagen: Der Auftragsbericht konnte nicht in Ihren Berichts-Bucket geschrieben werden.

  Dieser Fehler tritt auf, wenn die für den Batch-Operations-Auftrag verwendete IAM-Rolle den Abschlussbericht nicht an dem Speicherort speichern kann, der während der Auftragserstellung angegeben wurde. Überprüfen Sie zum Beheben dieses Fehlers, ob die IAM-Rolle über `PutObject`-Berechtigungen für den Bucket verfügt, in dem der Batch-Operations-Abschlussbericht gespeichert werden soll. Eine bewährte Methode besteht darin, den Bericht in einem anderen Bucket als dem Quell-Bucket bereitzustellen.
- Die Batchoperation wurde mit Fehlern abgeschlossen und die Gesamtzahl der Fehlschläge ist nicht 0.

  Dieser Fehler tritt auf, wenn bei dem ausgeführten Batchreplikationsauftrag unzureichende Objektberechtigungen vorhanden sind. Wenn Sie eine Replikationsregel für Ihren

Batchreplikationsauftrag verwenden, stellen Sie sicher, dass die für die Replikation verwendete IAM-Rolle über die richtigen Berechtigungen für den Zugriff auf Objekte aus dem Quell- oder Ziel-Bucket verfügt. Sie können auch den Abschlussbericht zur Batchreplikation anzeigen, um den spezifischen Grund für das Fehlschlagen der Replikation in Amazon S3 zu überprüfen.

- Der Batchauftrag wurde erfolgreich ausgeführt, die Anzahl der im Ziel-Bucket erwarteten Objekte stimmt nicht überein.

  Dieser Fehler tritt auf, wenn die im Manifest des Batchreplikationsauftrags aufgeführten Objekte nicht mit den Filtern übereinstimmen, die Sie während der Auftragserstellung ausgewählt haben. Möglicherweise erhalten Sie diese Meldung auch, wenn die Objekte in Ihrem Quell-Bucket keinen Replikationsregeln entsprechen und nicht im generierten Manifest enthalten sind.

Behebung von Fehlern bei der Server-Zugriffsprotokollierung

Die folgenden Themen können Ihnen bei der Behebung von Problemen helfen, die beim Einrichten der Protokollierung mit Amazon S3 auftreten können.

Themen

- Häufige Fehlermeldungen beim Einrichten der Protokollierung
- Behebung von Bereitstellungsfehlern

Häufige Fehlermeldungen beim Einrichten der Protokollierung

Die folgenden häufigen Fehlermeldungen können angezeigt werden, wenn Sie die Protokollierung über die AWS Command Line Interface (AWS CLI) und AWS-SDKs aktivieren:

Fehler: S3-standortübergreifende Protokollierung nicht zulässig

Wenn sich der Ziel-Bucket in einer anderen Region als der Quell-Bucket befindet, tritt der Fehler S3-standortübergreifende Protokollierung nicht zulässig auf. Um diesen Fehler zu beheben, stellen Sie sicher, dass sich der Ziel-Bucket, der für den Empfang der Zugriffsprotokolle konfiguriert ist, in derselben AWS-Region und demselben AWS-Konto wie der Quell-Bucket befindet.

Fehler: Die Eigentümer des zu protokollierenden Buckets und des Ziel-Buckets müssen identisch sein

Wenn Sie die Server-Zugriffsprotokollierung aktivieren, tritt dieser Fehler auf, wenn der angegebene Ziel-Bucket zu einem anderen Konto gehört. Um diesen Fehler zu beheben, stellen Sie sicher, dass sich der Ziel-Bucket in demselben AWS-Konto wie der Quell-Bucket befindet.

> **Note**
>
> Wir empfehlen, einen anderen Ziel-Bucket als den Quell-Bucket auszuwählen. Wenn Quell-Bucket und Ziel-Bucket identisch sind, werden für die Protokolle, die in den Bucket geschrieben werden, zusätzliche Protokolle erstellt, wodurch die Speicherkosten steigen können. Diese zusätzlichen Protokolle zu den Protokollen können es zudem schwierig machen, die gesuchten Protokolle zu finden. Zur einfacheren Protokollverwaltung empfehlen wir, Zugriffsprotokolle in einem anderen Bucket zu speichern. Weitere Informationen finden Sie unter the section called “Wie aktiviere ich die Protokollzustellung?”.

Fehler: Der Ziel-Bucket für die Protokollierung ist nicht vorhanden

Der Ziel-Bucket muss vorhanden sein, bevor die Konfiguration festgelegt wird. Dieser Fehler weist darauf hin, dass der Ziel-Bucket nicht vorhanden ist oder nicht gefunden werden kann. Vergewissern Sie sich, dass der Bucket-Name richtig geschrieben ist, und wiederholen Sie den Vorgang.

Fehler: Ziel-Erteilungen sind für Buckets mit „Bucket-Eigentümer erzwungen“ nicht zulässig

Dieser Fehler weist darauf hin, dass der Ziel-Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für die S3-Objekt-Eigentümerschaft verwendet. Die Einstellung „Vom Bucket-Eigentümer erzwungen“ unterstützt keine Ziel-Erteilungen. Weitere Informationen finden Sie unter Berechtigungen für Protokollbereitstellung.

Behebung von Bereitstellungsfehlern

Halten Sie sich an die folgenden bewährten Methoden, um Probleme mit der Server-Zugriffsprotokollierung zu vermeiden:

- Die S3-Protokollbereitstellungsgruppe hat Schreibzugriff auf den Ziel-Bucket – Die S3-Protokollbereitstellungsgruppe liefert Serverzugriffsprotokolle an den Ziel-Bucket. Es kann eine Bucket-Richtlinie oder eine Bucket-Zugriffssteuerungsliste (ACL) verwendet werden, um Schreibzugriff auf den Ziel-Bucket zu gewähren. Wir empfehlen jedoch, eine Bucket-Richtlinie anstelle einer ACL zu verwenden. Weitere Informationen zum Gewähren von Schreibzugriff für Ihren Ziel-Bucket finden Sie unter Berechtigungen für Protokollbereitstellung.

> **Note**
>
> Wenn der Ziel-Bucket die Einstellung „Vom Bucket-Eigentümer erzwungen“ für die Objekt-Eigentümerschaft verwendet, beachten Sie Folgendes:
>
> - ACLs sind deaktiviert und wirken sich nicht mehr auf die Berechtigungen aus. Somit können Sie Ihre Bucket-ACL nicht aktualisieren, um Zugriff auf die S3-Protokollbereitstellungsgruppe zu gewähren. Sie müssen vielmehr die Bucket-Richtlinie für den Ziel-Bucket aktualisieren, um Zugriff auf den Prinzipal des Protokollierungsservices zu gewähren.
> - Sie können keine Ziel-Erteilungen in Ihre `PutBucketLogging`-Konfiguration einschließen.

- Die Bucket-Richtlinie für den Ziel-Bucket lässt den Zugriff auf die Protokolle zu – Überprüfen Sie die Bucket-Richtlinie des Ziel-Buckets. Suchen Sie in der Bucket-Richtlinie nach allen Anweisungen, die `"Effect": "Deny"` enthalten. Stellen Sie anschließend sicher, dass die `Deny`-Anweisung das Schreiben von Zugriffsprotokollen in den Bucket nicht verhindert.
- S3 Object Lock ist auf dem Ziel-Bucket nicht aktiviert – Prüfen Sie, ob Object Lock für den Ziel-Bucket aktiviert ist. Die Objektsperre blockiert die Bereitstellung von Server-Zugriffsprotokollen. Sie müssen einen Ziel-Bucket auswählen, für den Object Lock nicht aktiviert ist.
- Von Amazon S3 verwaltete Schlüssel (SSE-S3) sind ausgewählt, wenn die Standardverschlüsselung auf dem Ziel-Bucket aktiviert ist – Sie können die Bucket-Standardverschlüsselung nur für den Ziel-Bucket verwenden, wenn Sie die serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verwenden. Die standardmäßige serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) wird für Ziel-Buckets für die Server-Zugriffsprotokollierung nicht unterstützt. Weitere Informationen zum Aktivieren der Standardverschlüsselung finden Sie unter Konfigurieren der Standardverschlüsselung.
- Für den Ziel-Bucket ist „Zahlung durch den Anforderer“ nicht aktiviert – Die Verwendung eines Buckets mit „Zahlung durch den Anforderer“ als Ziel-Bucket für die Server-Zugriffsprotokollierung wird nicht unterstützt. Um die Bereitstellung von Serverzugriffsprotokollen zu ermöglichen, deaktivieren Sie die Option „Zahlung durch den Anforderer“ im Ziel-Bucket.
- Überprüfen Sie Ihre AWS Organizations-Service-Kontrollrichtlinie – Überprüfen Sie bei Verwendung von AWS Organizations die Service-Kontrollrichtlinien, um sicherzustellen, dass der Amazon-S3-Zugriff zulässig ist. Service-Kontrollrichtlinien geben die maximalen Berechtigungen für die betroffenen Konten an. Suchen Sie in der Service-Kontrollrichtlinie nach Anweisungen, die

`"Effect": "Deny"` enthalten, und stellen Sie sicher, dass `Deny`-Anweisungen das Schreiben von Zugriffsprotokollen in den Bucket nicht verhindern. Weitere Informationen finden Sie unter Service-Kontrollrichtlinien (SCPs) im AWS Organizations-Benutzerhandbuch.

- Eine gewisse Zeit warten, bis die letzten Änderungen an der Protokollierungskonfiguration wirksam werden – Die erstmalige Aktivierung der Server-Zugriffsprotokollierung oder die Änderung des Ziel-Buckets für Protokolle wird erst nach einer gewissen Zeit vollständig wirksam. Es kann über eine Stunde dauern, bis alle Anforderungen ordnungsgemäß protokolliert und übermittelt wurden.

  Um nach Fehlern bei der Protokollzustellung zu suchen, aktivieren Sie Anforderungsmetriken in Amazon CloudWatch. Wenn die Protokolle nicht innerhalb weniger Stunden bereitgestellt werden, suchen Sie nach der Metrik `4xxErrors`, die auf Protokollbereitstellungsfehler hinweisen kann. Weitere Informationen zur Aktivierung von Anforderungsmetriken finden Sie unter the section called “Erstellen einer Metrik-Konfiguration für alle Objekte”.

Fehlerbehebung für die Versionsverwaltung

Die folgenden Themen können bei der Behebung von allgemeinen Problemen mit der Amazon-S3-Versionsverwaltung hilfreich sein.

Themen

- Ich möchte Objekte wiederherstellen, die in einem Bucket mit aktivierter Versionsverwaltung versehentlich gelöscht wurden.
- Ich möchte versionierte Objekte dauerhaft löschen
- Nach dem Aktivieren der Bucket-Versionsverwaltung stelle ich Leistungseinbußen fest

Ich möchte Objekte wiederherstellen, die in einem Bucket mit aktivierter Versionsverwaltung versehentlich gelöscht wurden.

Wenn Objektversionen aus S3-Buckets gelöscht werden, hat Amazon S3 im Allgemeinen keine Möglichkeit, diese wiederherzustellen. Wenn Sie jedoch die S3-Versionsverwaltung in Ihrem S3-Bucket aktiviert haben, kann ein Objekt mit einer `DELETE`-Anforderung, in der keine Versions-ID angegeben ist, nicht dauerhaft gelöscht werden. Stattdessen wird eine Löschmarkierung als Platzhalter hinzugefügt. Diese Löschmarkierung wird zur aktuellen Version des Objekts.

Gehen Sie wie folgt vor, um zu überprüfen, ob Ihre gelöschten Objekte dauerhaft oder vorübergehend (mit einer Löschmarkierung an ihrer Stelle) gelöscht wurden:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Wählen Sie in der Liste Buckets den Namen des Buckets aus, der das Objekt enthält.
4. Aktivieren Sie in der Liste Objekte den Schalter Versionen anzeigen rechts neben der Suchleiste und suchen Sie dann in der Suchleiste nach dem gelöschten Objekt. Dieser Schalter ist nur verfügbar, wenn die Versionsverwaltung zuvor für den Bucket aktiviert wurde.

   Sie können auch das S3 Inventory verwenden, um nach gelöschten Objekten zu suchen.
5. Wenn Sie das Objekt nicht finden können, nachdem Sie Versionen anzeigen eingeschaltet oder einen Bestandsbericht erstellt haben, und auch keine Löschmarkierung für das Objekt zu finden ist, ist der Löschvorgang dauerhaft und das Objekt kann nicht wiederhergestellt werden.

Sie können den Status eines gelöschten Objekts auch mit der API-Operation `HeadObject` über die AWS Command Line Interface (AWS CLI) prüfen. Verwenden Sie hierfür den folgenden Befehl `head-object` und ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen:

```
aws s3api head-object --bucket DOC-EXAMPLE-BUCKET --key index.html
```

Wenn Sie den Befehl `head-object` für ein versioniertes Objekt ausführen, dessen aktuelle Version eine Löschmarkierung ist, erhalten Sie den Fehler 404 Nicht gefunden. Beispielsweise:

Beim Aufrufen der HeadObject Operation ist ein Fehler aufgetreten (404): Nicht gefunden

Wenn Sie den Befehl `head-object` für ein versioniertes Objekt ausführen und die Versions-ID des Objekts angeben, ruft Amazon S3 die Metadaten des Objekts ab und bestätigt damit, dass das Objekt noch vorhanden ist und nicht dauerhaft gelöscht wurde.

```
aws s3api head-object --bucket DOC-EXAMPLE-BUCKET --key index.html --
version-id versionID
```

```
{
"AcceptRanges": "bytes",
"ContentType": "text/html",
"LastModified": "Thu, 16 Apr 2015 18:19:14 GMT",
"ContentLength": 77,
"VersionId": "Zg5HyL7m.eZU9iM7AVlJkrqAiE.0UG4q",
```

```
"ETag": "\"30a6ec7e1a9ad79c203d05a589c8b400\"",
"Metadata": {}
}
```

Wenn das Objekt gefunden wird und die neueste Version eine Löschmarkierung ist, ist die vorherige Version des Objekts immer noch vorhanden. Da die Löschmarkierung die aktuelle Version des Objekts darstellt, können Sie das Objekt wiederherstellen, indem Sie die Löschmarkierung löschen.

Nachdem Sie die Löschmarkierung dauerhaft entfernt haben, wird die zweitneueste Version des Objekts zur aktuellen Version des Objekts, sodass Ihr Objekt wieder verfügbar ist. Eine visuelle Darstellung der Wiederherstellung von Objekten finden Sie unter Entfernen von Löschmarkierungen.

Um eine bestimmte Version eines Objekts entfernen zu können, müssen Sie der Bucket-Eigentümer sein. Um eine Löschmarkierung dauerhaft zu löschen, müssen Sie Ihre Versions-ID in einer `DeleteObject`-Anforderung angeben. Verwenden Sie zum Löschen der Löschmarkierung den folgenden Befehl und ersetzen Sie die *`user input placeholders`* durch Ihre eigenen Informationen:

```
aws s3api delete-object --bucket DOC-EXAMPLE-BUCKET --key index.html --
version-id versionID
```

Weitere Informationen über den Befehl `delete-object` finden Sie unter delete-object in der AWS CLI-Befehlsreferenz. Weitere Informationen zum dauerhaften Löschen von Löschmarkierungen finden Sie unter Verwalten von Löschmarkierungen.

Ich möchte versionierte Objekte dauerhaft löschen

In einem Bucket mit aktivierter Versionsverwaltung kann ein Objekt mit einer `DELETE`-Anforderung ohne Versions-ID nicht dauerhaft gelöscht werden. Bei einer solchen Anforderung wird vielmehr eine Löschmarkierung eingefügt.

Um versionierte Objekte dauerhaft zu löschen, können Sie eine der folgenden Methoden auswählen:

- Erstellen Sie eine S3-Lebenszyklusregel, um nicht aktuelle Versionen dauerhaft zu löschen. Um nicht aktuelle Versionen dauerhaft zu löschen, wählen Sie Nicht aktuelle Versionen von Objekten dauerhaft löschen aus und geben Sie anschließend eine Zahl unter Tage, nachdem Objekte veraltet sind ein. Sie können die Anzahl der beizubehaltenden neueren Versionen optional angeben, indem Sie einen Wert unter Number of newer versions to retain (Anzahl der beizubehaltenden neueren Versionen) eingeben. Weitere Informationen zum Erstellen dieser Regel finden Sie unter Festlegen einer S3-Lebenszyklus-Konfiguration.

- Löschen Sie eine angegebene Version, indem Sie die Versions-ID in die DELETE-Anforderung aufnehmen. Weitere Informationen hierzu finden Sie unter Dauerhaftes Löschen von versionierten Objekten.
- Erstellen Sie eine Lebenszyklusregel, um aktuelle Versionen ablaufen zu lassen. Um aktuelle Versionen von Objekten ablaufen zu lassen, wählen Sie Aktuelle Objektversionen ablaufen lassen aus und fügen Sie dann eine Zahl unter Tage nach Objekterstellung ein. Weitere Informationen zum Erstellen dieser Lebenszyklusregel finden Sie unter Festlegen einer S3-Lebenszyklus-Konfiguration.
- Um alle versionierten Objekte und Löschmarkierungen dauerhaft zu löschen, erstellen Sie zwei Lebenszyklusregeln: eine, um aktuelle Versionen ablaufen zu lassen und nicht aktuelle Versionen von Objekten dauerhaft zu löschen, und die andere, um Löschmarkierungen für abgelaufene Objekte zu löschen.

In einem Bucket mit aktivierter Versionsverwaltung können mit einer DELETE-Anforderung, in der keine Versions-ID angegeben ist, nur Objekte mit einer Versions-ID NULL entfernt werden. Wenn das Objekt hochgeladen wurde, als die Versionsverwaltung aktiviert war, wird mit einer DELETE-Anforderung, in der keine Versions-ID angegeben ist, eine Löschmarkierung für dieses Objekt erstellt.

> **Note**
>
> Bei Buckets mit aktivierter S3-Objektsperre verursacht eine DELETE-Objektanforderung mit einer ID einer geschützten Objektversion den Fehler 403 Zugriff verweigert. Eine DELETE-Objektanforderung ohne Versions-ID fügt eine Löschmarkierung als neueste Version des Objekts mit der Antwort 200 OK hinzu. Objekte, die durch die Objektsperre geschützt sind, können erst dauerhaft gelöscht werden, wenn ihre Aufbewahrungszeiträume und gesetzlichen Aufbewahrungsfristen aufgehoben sind. Weitere Informationen finden Sie unter the section called “So funktioniert die S3-Objektsperre”.

Nach dem Aktivieren der Bucket-Versionsverwaltung stelle ich Leistungseinbußen fest

Bei Buckets mit aktivierter Versionsverwaltung kann es zu Leistungseinbußen kommen, wenn zu viele Löschmarkierungen oder versionierte Objekte vorhanden sind und wenn bewährte Methoden nicht befolgt werden.

Zu viele Löschmarkierungen

Nachdem Sie die Versionsverwaltung für einen Bucket aktiviert haben, wird mit einer DELETE-Anforderung ohne Versions-ID an ein Objekt eine Löschmarkierung mit einer eindeutigen Versions-ID erstellt. Lebenszyklus-Konfigurationen mit der Regel Aktuelle Objektversionen ablaufen lassen fügen jedem Objekt eine Löschmarkierung mit einer eindeutigen Versions-ID hinzu. Wenn zu viele Löschmarkierungen vorhanden sind, kann dies die Leistung im Bucket beeinträchtigen.

Wenn die Versionsverwaltung für einen Bucket ausgesetzt wird, markiert Amazon S3 die Versions-ID bei neu erstellten Objekten als NULL. Eine Ablaufaktion in einem Bucket mit ausgesetzter Versionsverwaltung bewirkt, dass Amazon S3 eine Löschmarkierung mit der Versions-ID NULL erstellt. In einem Bucket mit ausgesetzter Versionsverwaltung wird für jede Löschanforderung eine NULL-Löschmarkierung erstellt. Diese NULL-Löschmarkierungen werden auch als Löschmarkierungen für abgelaufenes Objekt bezeichnet, wenn alle Objektversionen gelöscht werden und nur eine einzelne Löschmarkierung übrig bleibt. Wenn sich zu viele NULL-Löschmarkierungen ansammeln, kommt es zu Leistungseinbußen im Bucket.

Zu viele versionierte Objekte

Wenn ein Bucket mit aktivierter Versionsverwaltung Objekte mit Millionen von Versionen enthält, kann es vermehrt zu Fehlern 503 Service nicht verfügbar kommen. Wenn Sie eine deutliche Zunahme der HTTP-Antworten 503 Service Unavailable feststellen, die für PUT- oder DELETE-Objektanforderungen an einen Bucket mit aktivierter Versionsverwaltung eingehen, befinden sich möglicherweise ein oder mehrere Objekte in dem Bucket, für die Millionen von Versionen vorhanden sind. Wenn Sie Objekte mit Millionen Versionen haben, drosselt Amazon S3 Anforderungen an den Bucket automatisch. Durch die Drosselung von Anforderungen wird Ihr Bucket vor übermäßigem Anforderungsdatenverkehr geschützt, der andere Anforderungen an denselben Bucket verhindern könnte.

Um festzustellen, zu welchen Objekten es Millionen Versionen gibt, verwenden Sie S3 Inventory. S3 Inventory generiert einen Bericht, der eine flache Dateiliste der Objekte in einem Bucket enthält. Weitere Informationen finden Sie unter Amazon S3 Inventory.

Um zu überprüfen, ob der Bucket viele versionierte Objekte enthält, verwenden Sie die S3-Storage-Lens-Metriken, um die Anzahl der Objekte der aktuellen Version, die Anzahl nicht aktueller Versionsobjekte und die Anzahl der Löschmarkierungsobjekte anzuzeigen. Weitere Informationen zu Storage-Lens-Metriken finden Sie unter Amazon S3-Storage-Lens-Metrikglossar.

Das Amazon-S3-Team fordert die Kunden auf, Anwendungen zu überprüfen, die wiederholt dasselbe Objekt überschreiben und damit potenziell Millionen Versionen für dieses Objekt erstellen, um festzustellen, ob die Anwendung wie beabsichtigt funktioniert. Beispielsweise kann eine Anwendung,

die eine Woche lang jede Minute dasselbe Objekt überschreibt, über zehntausend Versionen erstellen. Wir empfehlen, für jedes Objekt weniger als einhunderttausend Versionen zu speichern. Wenn Sie einen Anwendungsfall haben, der Millionen von Versionen für ein oder mehrere Objekte erfordert, wenden Sie sich an das AWS Support-Team, um mit dessen Unterstützung eine bessere Lösung zu finden.

Bewährte Methoden

Um Leistungseinbußen im Zusammenhang mit der Versionsverwaltung zu vermeiden, empfehlen wir Ihnen, den folgenden bewährten Methoden zu folgen:

- Aktivieren Sie eine Lebenszyklusregel, um die vorherigen Versionen von Objekten ablaufen zu lassen. Sie können beispielsweise eine Lebenszyklusregel erstellen, nach der nicht aktuelle Versionen ablaufen, wenn das Objekt 30 Tage nicht mehr aktuell ist. Sie können auch mehrere nicht aktuelle Versionen beibehalten, wenn Sie nicht alle löschen möchten. Weitere Informationen finden Sie unter Festlegen einer S3-Lebenszyklus-Konfiguration.
- Aktivieren Sie eine Lebenszyklusregel, um Löschmarkierungen für abgelaufene Objekte zu löschen, denen keine Datenobjekte im Bucket zugeordnet sind. Weitere Informationen finden Sie unter Löschen abgelaufener Löschmarkierungen für Objekte.

Weitere bewährte Methoden zur Amazon-S3-Leistungsoptimierung finden Sie unter Bewährte Methoden für Designmuster.

Abrufen der Anforderungs-IDs in Amazon S3 für AWS Support

Wenn Sie mit dem AWS Support in Kontakt treten, weil Sie Fehler oder ein unerwartetes Verhalten in Amazon S3 festgestellt haben, müssen Sie die Anforderungs-ID der fehlgeschlagenen Aktion bereitstellen. Der AWS Support verwendet diese Anforderungs-IDs, um Sie bei der Lösung der aufgetretenen Probleme zu unterstützen.

Anfrage-IDs werden paarweise vergeben. Sie werden in jeder von Amazon S3 verarbeiteten Antwort zurückgegeben (auch in den fehlerhaften). Der Zugriff darauf erfolgt über Verbose-Protokolle. Es gibt verschiedene übliche Verfahren zum Abrufen der Anforderungs-IDs, darunter S3-Zugriffsprotokolle und AWS CloudTrail-Ereignisse oder -Datenereignisse.

Nachdem Sie diese Protokolle wiederhergestellt haben, kopieren Sie diese beiden Werte. Sie brauchen sie, wenn Sie den AWS Support kontaktieren. Weitere Informationen zur Kontaktaufnahme

mit dem AWS Support finden Sie unter AWS kontaktieren und in der Dokumentation zum AWS Support.

Abrufen der Anforderungs-IDs mithilfe von HTTP

Sie können Ihre Anfrage-IDs `x-amz-request-id` und `x-amz-id-2` ermitteln, indem Sie die Abschnitte einer HTTP-Anfrage protokollieren, bevor sie die Zielanwendung erreicht. Es gibt verschiedene Tools von Drittanbietern, mit denen Verbose-Protokolle für HTTP-Anfragen wiederhergestellt werden können. Wählen Sie ein Tool, dem Sie vertrauen, führen Sie es aus und überwachen Sie den Port, über den Ihr Amazon-S3-Datenverkehr läuft, während Sie eine weitere Amazon-S3-HTTP-Anforderung senden.

Bei HTTP-Anforderungen sieht das Paar der Anforderungs-IDs wie in den folgenden Beispielen gezeigt aus:

```
x-amz-request-id: 79104EXAMPLEB723
x-amz-id-2: IOWQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km
```

> **Note**
>
> HTTPS-Anfragen werden in den meisten Paketerfassungen verschlüsselt und verborgen.

Abrufen der Anforderungs-IDs mithilfe eines Webbrowsers

Die meisten Webbrowser beinhalten Entwickler-Tools, mit denen Sie Anforderungs-Header anzeigen können.

Für auf einem Webbrowser basierte Anfragen, die einen Fehler zurückgeben, sieht das Paar der Anfrage-IDs wie in den folgenden Beispielen gezeigt aus.

```
<Error><Code>AccessDenied</Code><Message>Access Denied</Message>
<RequestId>79104EXAMPLEB723</RequestId><HostId>IOWQ4fDEXAMPLEQM
+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km</HostId></Error>
```

Zeigen Sie die HTTP-Anforderungs-Header mithilfe der Entwickler-Tools zu Ihrem Browser an, um das Paar der Anforderungs-IDs aus erfolgreichen Anforderungen abzurufen. Weitere Informationen zu Entwickler-Tools für spezifische Browser finden Sie unter Fehlerbehebung für Amazon S3 – Wiederherstellung Ihrer S3-Anforderungs-IDs in AWS re:Post.

Abrufen der Anforderungs-IDs mithilfe der AWS-SDKs

Die folgenden Abschnitte enthalten Informationen für die Konfiguration der Protokollierung unter Verwendung eines AWS-SDK. Auch wenn Sie die Verbose-Protokollierung für jede Anforderung und jede Antwort aktivieren können, sollten Sie die Protokollierung nicht in Produktionssystemen aktivieren, da große Anforderungen und Antworten eine Anwendung erheblich verlangsamen können.

Für AWS-SDK-Anfragen sieht das Paar der Anfrage-IDs wie in den folgenden Beispielen gezeigt aus.

```
Status Code: 403, AWS Service: Amazon S3, AWS Request ID: 79104EXAMPLEB723
AWS Error Code: AccessDenied  AWS Error Message: Access Denied
S3 Extended Request ID: IOWQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK
+Jd1vEXAMPLEa3Km
```

Abrufen der Anforderungs-IDs mithilfe des SDK für PHP

Sie können die Protokollierung mit PHP konfigurieren. Weitere Informationen finden Sie unter Wie kann ich sehen, welche Daten übertragen wurden? im Entwicklerhandbuch zum AWS SDK for PHP.

Abrufen der Anforderungs-IDs mithilfe des SDK für Java

Sie können die Protokollierung für spezifische Anforderungen oder Antworten aktivieren, um nur die relevanten Header zu erfassen und zurückzugeben. Dazu importieren Sie die Klasse `com.amazonaws.services.s3.S3ResponseMetadata`. Anschließend können Sie die Anforderung in einer Variablen speichern, bevor Sie die eigentliche Anforderung ausführen. Rufen Sie `getCachedResponseMetadata(AmazonWebServiceRequest request).getRequestID()` auf, um die protokollierte Anforderung oder Antwort zu erhalten.

Example

```
PutObjectRequest req = new PutObjectRequest(bucketName, key, createSampleFile());
s3.putObject(req);
S3ResponseMetadata md = s3.getCachedResponseMetadata(req);
System.out.println("Host ID: " + md.getHostId() + " RequestID: " + md.getRequestId());
```

Alternativ können Sie eine Verbose-Protokollierung jeder Java-Anfrage und -Antwort verwenden. Weitere Informationen finden Sie unter Verbose-Protokollierung des Netzwerkverkehrs im Entwicklerhandbuch zu AWS SDK for Java.

Abrufen der Anforderungs-IDs mithilfe des AWS SDK for .NET

Mit dem integrierten AWS SDK for .NET-Protokollierungs-Tool können Sie die Protokollierung mithilfe des `System.Diagnostics` konfigurieren. Weitere Informationen finden Sie im Beitrag Protokollierung mit dem AWS SDK für .NET im AWS-Blog für Entwickler.

> **Note**
>
> Standardmäßig enthält das zurückgegebene Protokoll nur Fehlerinformationen. Zum Abrufen der Anforderungs-IDs muss der Konfigurationsdatei `AWSLogMetrics` (und optional `AWSResponseLogging`) hinzugefügt werden.

Abrufen der Anforderungs-IDs mithilfe des SDK für Python (Boto3)

Mit dem AWS SDK for Python (Boto3) können Sie spezifische Antworten protokollieren. Sie können diese Funktion verwenden, um nur die relevanten Header zu erfassen. Der folgende Code zeigt, wie Teile der Antwort in einer Datei protokolliert werden:

```
import logging
import boto3
logging.basicConfig(filename='logfile.txt', level=logging.INFO)
logger = logging.getLogger(__name__)
s3 = boto3.resource('s3')
response = s3.Bucket(bucket_name).Object(object_key).put()
logger.info("HTTPStatusCode: %s", response['ResponseMetadata']['HTTPStatusCode'])
logger.info("RequestId: %s", response['ResponseMetadata']['RequestId'])
logger.info("HostId: %s", response['ResponseMetadata']['HostId'])
logger.info("Date: %s", response['ResponseMetadata']['HTTPHeaders']['date'])
```

Sie können auch Ausnahmen abfangen und relevante Informationen protokollieren, wenn eine Ausnahme ausgelöst wird. Weitere Informationen finden Sie unter Erkennen nützlicher Informationen in Fehlerantworten in der API-Referenz zum AWS SDK für Python (Boto).

Darüber hinaus können Sie Boto3 mit dem folgenden Code für die Ausgabe ausführlicher Debugging-Protokolle konfigurieren:

```
import boto3
boto3.set_stream_logger('', logging.DEBUG)
```

Weitere Informationen finden Sie unter set_stream_logger in der API-Referenz zum AWS SDK für Python (Boto).

Abrufen der Anforderungs-IDs mithilfe des SDK für Ruby

Sie können die Anforderungs-IDs mit dem SDK für Ruby Version 1, 2 oder 3 abrufen.

- Verwenden von SDK für Ruby – Version 1 – Mit der folgenden Codezeile können Sie eine globale HTTP-Übertragungsprotokollierung aktivieren.

```
s3 = AWS::S3.new(:logger => Logger.new($stdout), :http_wire_trace => true)
```

- Verwenden von SDK für Ruby – Version 2 oder Version 3 – Mit der folgenden Codezeile können Sie eine globale HTTP-Übertragungsprotokollierung aktivieren.

```
s3 = Aws::S3::Client.new(:logger => Logger.new($stdout), :http_wire_trace => true)
```

Tipps zum Abrufen von Übertragungsinformationen von einem AWS-Client finden Sie unter Tipp zum Debuggen: Abrufen von Übertragungsnachverfolgungsinformationen von einem Client.

Abrufen der Anforderungs-IDs mithilfe der AWS CLI

Fügen Sie zum Abrufen der Anforderungs-IDs bei Verwendung der AWS Command Line Interface (AWS CLI) Ihrem Befehl `--debug` hinzu.

Abrufen der Anforderungs-IDs mithilfe von Windows PowerShell

Informationen zum Wiederherstellen von Protokollen mit Windows PowerShell finden Sie im Beitrag Antwort-Protokollierung in AWS Tools for Windows PowerShell im .NET-Entwicklungsblog.

Abrufen der Anforderungs-IDs mithilfe von AWS CloudTrail-Datenereignissen

Ein Amazon-S3-Bucket, der mit CloudTrail-Datenereignissen konfiguriert ist, um S3-API-Operationen auf Objektebene zu protokollieren, stellt detaillierte Informationen zu Aktionen bereit, die von einem Benutzer, einer Rolle oder einem AWS-Service in Amazon S3 ausgeführt werden. Sie können S3-Anforderungs-IDs identifizieren, indem Sie CloudTrail-Ereignisse mit Athena abfragen.

Abrufen der Anforderungs-IDs mithilfe der S3-Server-Zugriffsprotokollierung

Ein Amazon-S3-Bucket, der für die S3-Server-Zugriffsprotokollierung konfiguriert ist, stellt detaillierte Aufzeichnungen zu allen an den Bucket gesendeten Anforderungen bereit. Sie können S3-Anforderungs-IDs identifizieren, indem Sie die Serverzugriffsprotokolle mit Athena abfragen.

Dokumentverlauf

- Aktuelle API-Version: 2006-03-01

In der folgenden Tabelle werden die wichtigen Änderungen in jeder Version der Amazon Simple Storage Service API-Referenz und des Amazon-S3-Benutzerhandbuchs beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

Änderung	Beschreibung	Datum
Amazon S3 Inventory unterstützt den Schlüssel s3:InventoryAccessibleOptionalFields condition	Amazon S3 Inventory unterstützt den s3:InventoryAccessibleOptionalFields condition -Schlüssel, um zu steuern, ob Benutzer optionale Metadatenfelder in ihre Berichte aufnehmen können. Weitere Informationen finden Sie unter Steuern der Erstellung von S3-Inventory-Berichten.	20. Februar 2024
IPv6-Unterstützung für S3 on Outposts	Sie können jetzt mit IPv6 über S3-on-Outposts-Dual-Stack-Endpunkte auf S3-on-Outposts-Buckets zugreifen. Die IPv6-Unterstützung für S3 on Outposts ermöglicht es Ihnen, Ihre S3-on-Outposts-Buckets zu verwalten und Ressourcen der Steuerebene über IPv6-Netzwerke zu steuern.	16. Januar 2024
Neue leistungsstarke Amazon-S3-Speicherklasse mit einer Zone – S3 Express One Zone	Amazon S3 Express One Zone ist eine leistungsstarke Amazon-S3-Speicherklasse	28. November 2023

	mit einer einzelnen Zone, die speziell für den konsisten ten Datenzugriff im einstelli gen Millisekundenbereich für Ihre latenzempfindlichs ten Anwendungen entwickelt wurde. Weitere Informationen finden Sie unter S3 Express One Zone.	
Mountpoint für Amazon S3 fügt Unterstützung für S3 Express One Zone hinzu	Sie können jetzt S3-Express-One-Zone-Verzeichnis-Buck ets mit Mountpoint mounten.	28. November 2023
Lambda-Aufrufschemaversion	Amazon S3 Batch Operation s führt eine neue Lambda-Aufrufschemaversion für Batch-Operations-Aufträge ein, die auf Verzeichnis-Buckets ausgeführt werden. Verwenden von Lambda und Amazon S3 Batch Operations mit Verzeichnis-Buckets.	28. November 2023
Importaktion für Verzeichnis-Buckets	Amazon S3 führt die Importakt ion ein. Import ist eine optimierte Methode zur Erstellung von Amazon-S3-Batch-Operations-Aufträgen zum Kopieren von Objekten aus Allzweck-Buckets in Verzeichnis-Buckets. Weitere Informationen finden Sie unter Importieren von Objekten in einen Verzeichnis-Bucket .	28. November 2023

Verwalten des S3-Zugriffs mit S3 Access Grants	Mit Amazon S3 Access Grants können Sie Datenberechtigungen für AWS Identity and Access Management (IAM)-Prinzipale zusätzlich zu Verzeichnisidentitäten aus Unternehmensverzeichnissen wie verwaltenAzure AD. Sie können jetzt S3-Berechtigungen mit den geringsten Berechtigungen durchsetzen und diese Berechtigungen ganz einfach an Ihre geschäftlichen Anforderungen anpassen. Weitere Informationen finden Sie unter Verwalten des Datenzugriffs mit S3 Access Grants.	26. November 2023
Mountpoint für Amazon S3 fügt Caching-Feature hinzu	Mit Mountpoint können Sie jetzt das Caching für Daten konfigurieren, auf die wiederholt zugegriffen wird.	22. November 2023

Verbesserte Amazon S3-Batchoperationenmanifestgenerierung	Sie können Amazon S3 Batch Operations jetzt anweisen, automatisch ein Manifest auf der Grundlage von Objektfilterkriterien zu generieren, die Sie bei der Erstellung Ihres Auftrags angeben. Diese Option ist für Batch-Replikationsaufträge verfügbar, die Sie in der Amazon S3-Konsole erstellen, oder für jeden Auftragstyp, den Sie mithilfe der AWS CLI, AWS SDKs oder der Amazon S3-REST-API erstellen. Weitere Informationen finden Sie unter Erstellen eines Amazon-S3-Batch-Operations-Auftrags.	22. November 2023
Bestehende Amazon-S3-Buckets können jetzt Object-Lock-Konfigurationen hinzufügen	Sie können Object Lock jetzt für einen vorhandenen Amazon-S3-Bucket aktivieren. Sie können gesetzliche und andere Aufbewahrungsfristen für neue oder bestehende Buckets festlegen. Weitere Informationen finden Sie unter Verwenden von Object Lock.	20. November 2023
S3-Storage-Lens-Anforderungsmetriken für Präfixe	In S3 Storage Lens werden Anforderungsmetriken für Präfixe in einem Amazon-S3-Bucket eingeführt. Weitere Informationen finden Sie unter Metrikkategorien.	17. November 2023

Amazon-S3-Storage-Lens-Gruppen	In S3 Storage Lens werden Storage-Lens-Gruppen eingeführt. Hierbei handelt es sich um benutzerdefinierte Filter für Objekte auf der Grundlage von Objektmetadaten. Weitere Informationen finden Sie unter Arbeiten mit S3-Storage-Lens-Gruppen.	15. November 2023
Neue IAM-Richtlinie	S3 on Outposts führt die serviceverknüpfte Rolle `AWSServiceRoleForS3nOutposts` ein, die Sie bei der Verwaltung von Netzwerkressourcen unterstützt. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für S3 on Outposts.	3. Oktober 2023
Amazon S3 bietet den `Last-Modified` -Zeitpunkt für das Löschen von Markierungen	Amazon S3 stellt den `Last-Modified` -Zeitpunkt der Löschmarkierungen in den Antwort-Headern von S3 `Head`- und `Get`-API-Vorgängen bereit. Weitere Informationen finden Sie unter Arbeiten mit Löschmarkierungen.	27. September 2023

Amazon S3-Aktualisierung auf von AWS verwaltete Richtlinien	Amazon S3 hat die `s3:Describe*` -Berechtigungen zu `AmazonS3ReadOnlyAccess` hinzugefügt. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für Amazon S3.	11. August 2023
Verbesserte Startzeiten für Standard-Wiederherstellungsanfragen, die über S3 Batch Operations gestellt wurden	Standardabrufe für Wiederherstellungsanfragen, die über S3 Batch Operations gestellt wurden, können jetzt innerhalb von Minuten beginnen. Weitere Informationen finden Sie unter Optionen für den Archivabruf.	9. August 2023
Mountpoint, ein Client mit hohem Durchsatz zum Mounten eines Amazon-S3-Buckets als lokales Dateisystem wurde hinzugefügt.	Mit Mountpoint können Ihre Anwendungen über Dateioperationen auf die in Amazon S3 gespeicherten Objekte zugreifen, sodass Ihre Anwendungen über eine Dateischnittstelle auf den elastischen Speicher und den Durchsatz von Amazon S3 zugreifen können.	9. August 2023

Serverseitige Dual-Layer-Verschlüsselung mit - AWS Key Management Service Schlüsseln (DSSE-KMS)	Die serverseitige Dual-Layer-Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (DSSE-KMS) wendet zwei Verschlüsselungsebenen auf Objekte an, wenn sie in Amazon S3 hochgeladen werden. Weitere Informationen finden Sie unter Verwenden der serverseitigen Dual-Layer-Verschlüsselung mit AWS KMS Schlüsseln .	13. Juni 2023
Amazon S3 aktiviert S3 Block Public Access und deaktiviert S3-Zugriffssteuerungslisten (ACLs) für alle neuen Buckets.	Amazon S3 aktiviert jetzt automatisch S3 Block Public Access und deaktiviert S3-Zugriffskontrolllisten (ACLs) für alle neuen S3-Buckets in allen AWS Regionen. Weitere Informationen finden Sie unter Blockieren des öffentlichen Zugriffs auf Ihren Amazon-S3-Speicher und Steuern der Objekteigentümerschaft und Deaktivieren von ACLs für Ihren Bucket.	27. April 2023
Metrik zu fehlgeschlagenen S3-Replikationsvorängen	Amazon S3 fügt eine neue Amazon CloudWatch Metrik zur Überwachung von S3-Replikationsfehlern hinzu. Weitere Informationen finden Sie unter Überwachen des Fortschritts mit Replikationsmetriken.	05. April 2023

Privates DNS	AWS PrivateLink für Amazon S3 unterstützt jetzt Private DNS. Weitere Informationen finden Sie unter Privates DNS.	14. März 2023
Support für kontoüber greifende Zugriffspunkte in der Amazon-S3-Konsole	Amazon S3 unterstützt jetzt die Erstellung von kontoüber greifenden Zugriffspunkten mit der Amazon-S3-Konsole. Weitere Informationen finden Sie unter Erstellen von Zugriffspunkten.	14. März 2023
Amazon S3 in Outposts unterstützt S3 Replication in Outposts	Mit der lokalen S3-Replikation können Sie Objekte automatisch in einen einzelnen Ziel-Bucket oder in mehrere Ziel-Buckets von Outposts replizieren. Die Ziel-Buckets können sich in anderen AWS Outposts oder innerhalb derselben Outposts wie der Quell-Bucket befinden. Weitere Informationen finden Sie unter Replikation von Objekten für S3 in Outposts.	14. März 2023

Access-Point-Alias in Amazon S3 Object Lambda	Wenn Sie einen Object Lambda Access Point erstellen , generiert Amazon S3 automatisch einen eindeutig en Alias für den Object Lambda Access Point. Sie können diesen Alias anstelle eines Amazon-S3-Bucketna mens oder des Amazon-Ressourcennamens (ARN) des Object Lambda Access Point in einer Anforderung für Zugriffspunkt-Operationen auf Datenebene verwenden. Weitere Informationen finden Sie unter Verwenden eines Alias im Bucket-Stil für Ihren Object Lambda Access Point.	14. März 2023
Kontoübergreifender Support für Multi-Region Access Points in Amazon S3	Amazon S3 unterstützt jetzt die Erstellung von kontoübergreifenden Multi-Region Access Points mit der Amazon-S3-Konsole. Weitere Informationen finden Sie unter Erstellen von Multi-Region Access Points.	14. März 2023

Kontoübergreifende Zugriffspunkte	Amazon S3 unterstützt das Erstellen von kontoüber greifenden Zugriffspunkten. Sie können einen kontoüber greifenden Zugriffspunkt erstellen, indem Sie die AWS Command Line Interface (AWS CLI) oder die `CreateAccessPoint` - Operation der REST-API verwenden. Weitere Informati onen finden Sie unter Erstellen von Zugriffspunkten.	30. November 2022
Amazon S3 unterstützt Failover-Kontrollen für Amazon S3 Multi-Region Access Points	Amazon S3 führt Failover-Kontrollen für Multi-Reg ion Access Points ein. Mit diesen Kontrollen können Sie den S3-Datenzugriffsan forderungsverkehr, der über einen Amazon S3 Multi-Reg ion Access Point weitergel eitet wird, innerhalb weniger Minuten auf eine alternative AWS-Region umstellen, um hochverfügbare Anwendung en zu testen und zu erstellen. Weitere Informationen finden Sie unter Failover-Kontrollen für Amazon S3 Multi-Region Access Points.	28. November 2022

Amazon S3 Storage Lens erhöht die unternehmensweite Sichtbarkeit mit 34 neuen Metriken	S3 Storage Lens führt 34 zusätzliche Metriken ein, um umfassende Möglichkeiten zur Kostenoptimierung aufzudecken, bewährte Methoden für den Datenschutz zu identifizieren und die Leistung von Anwendungsworkflows zu verbessern. Weitere Informationen finden Sie unter Metriken von S3 Storage Lens.	17. November 2022
Amazon S3 unterstützt höhere Geschwindigkeiten für Wiederherstellungsanforderungen für S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive	Amazon S3 unterstützt Wiederherstellungsanforderungen mit einer Geschwindigkeit von bis zu 1 000 Transaktionen pro Sekunde, pro AWS-Konto für die Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive.	15. November 2022

Amazon S3 on Outposts unterstützt zusätzliche S3-Lebenszyklusaktionen und -filter	S3 on Outposts unterstützt zusätzliche S3-Lebenszyklusregeln zur Optimierung des Kapazitätsmanagements. Sie können Objekte ablaufen lassen, wenn sie veralten oder durch neuere Versionen ersetzt werden. Sie können eine Lebenszyklusregel für einen ganzen Bucket oder eine Teilmenge von Objekten in einem Bucket erstellen, indem Sie nach Präfixen, Objekt-Tags oder Objektgröße filtern. Weitere Informationen finden Sie unter Erstellen und Verwalten einer Lebenszyklus-Konfiguration.	2. November 2022
S3-Replication-Unterstützung für SSE-C-Objekte	Sie können Objekte replizieren, die mit der serversei tigen Verschlüsselung unter Verwendung der vom Kunden bereitgestellten Schlüssel erstellt wurden. Weitere Informationen zum Replizieren verschlüsselter Objekte finden Sie unter Replizieren von Objekten, die mit serversei tiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS) erstellt wurden.	24. Oktober 2022

Amazon S3 on Outposts unterstützt Zugriffspunkt-Aliasse	Bei S3 on Outposts müssen Sie Zugriffspunkte verwenden, um auf ein Objekt in einem Outposts-Bucket zuzugreifen. Jedes Mal, wenn Sie einen Zugriffspunkt für einen Bucket erstellen, generiert S3 on Outposts automatisch einen Zugriffspunkt-Alias. Sie können diesen Zugriffspunkt-Alias anstelle eines Zugriffspunkt-ARNs für jede Datenebenen-Operation verwenden. Weitere Informationen finden Sie unter Verwenden eines Alias im Bucket-Stil für den Zugriffspunkt Ihres S3-on-Outposts-Buckets.	21. Oktober 2022
S3-Objekt-Lambda unterstützt die Operationen `HeadObject`, `ListObjects` und `ListObjectsV2`	Sie können benutzerdefinierten Code verwenden, um die von den S3-Standardanforderungen `GET`, `LIST` oder `HEAD` zurückgegebenen Daten zu ändern, um Zeilen zu filtern, die Größe von Images dynamisch anzupassen, vertrauliche Daten zu unterdrücken und vieles mehr. Weitere Informationen finden Sie unter Transformieren von Objekten mit S3-Objekt Lambda.	4. Oktober 2022

Amazon S3 on Outposts unterstützt die S3-Versionsverwaltung	Wenn diese Option aktiviert ist, speichert die S3-Versionsverwaltung mehrere unterschiedliche Kopien eines Objekts im selben Bucket. Sie können die S3-Versionsverwaltung verwenden, um sämtliche Versionen aller Objekte in Ihren Outposts-Buckets zu speichern, abzurufen oder wiederherzustellen. Mit der S3-Versionsverwaltung können Sie Versionen sowohl nach unbeabsichtigten Benutzeraktionen als auch nach Anwendungsfehlern problemlos wiederherstellen. Weitere Informationen finden Sie unter Verwalten der S3-Versionsverwaltung für Ihren S3-on-Outposts-Bucket.	21. September 2022
AWS Backup für Amazon S3	AWS Backup ist ein vollständig verwalteter, richtlinienbasierter Service, mit dem Sie eine zentrale Backup-Richtlinie zum Schutz Ihrer Amazon S3-Daten definieren können. Weitere Informationen finden Sie unter Verwenden von AWS Backup für Amazon S3.	18. Februar 2022

Verwenden der S3-Batchreplikation, um bestehende Objekte zu replizieren	Mit der S3-Batchreplikation können Sie Objekte replizieren, die vor der Einführung einer Replikationskonfiguration existierten. Das Replizieren vorhandener Objekte erfolgt mithilfe eines Auftrags für Batchoperationen. Die S3-Batchreplikation unterscheidet sich von der Live-Replikation, die kontinuierlich und automatisch neue Objekte über Amazon-S3-Buckets hinweg kopiert. Weitere Informationen finden Sie unter Replizieren vorhandener Objekte mit S3-Batch-Replikation.	8. Februar 2022
Umbenennen von S3 Glacier Flexible Retrieval	Die Glacier-Speicherklasse wurde in S3 Glacier Flexible Retrieval umbenannt. Diese Änderung beeinträchtigt nicht die API.	30. November 2021

Neue S3-Objekt-Ownership-Einstellung zum Deaktivieren von ACLs	Sie können die erzwungene Einstellung für den Bucket-Eigentümer für Object Ownership anwenden, um ACLs für Ihren Bucket und die darin befindlichen Objekte zu deaktivieren und das Eigentum an jedem Objekt in Ihrem Bucket zu übernehme n. Die vom Bucket-Eigentümer erzwungene Einstellung vereinfacht die Zugriffsv erwaltung für in Amazon S3 gespeicherte Daten. Weitere Informationen finden Sie unter Steuern des Eigentums an Objekten und Deaktivieren von ACLs für Ihren Bucket.	30. November 2021
Neue S3-Intelligent-Tiering-Speicherklasse	S3 Intelligent-Tiering Archive Instant Access ist eine zusätzliche Speicherklasse unter S3 Intelligent-Tiering. Weitere Informationen finden Sie unter Wie funktioniert S3 Intelligent-Tiering.	30. November 2021
Neue Speicherklasse S3 Glacier Instant Retrieval	Sie können jetzt Objekte in der Speicherklasse S3 Glacier Instant Retrieval platziere n. Weitere Informationen zu dieser Speicherklasse finden Sie unter Verwenden von Amazon-S3-Speicherklassen.	30. November 2021

AWS Backup für Amazon S3 Preview	AWS Backup ist ein vollständig verwalteter, richtlinienbasierter Service, mit dem Sie eine zentrale Backup-Richtlinie zum Schutz Ihrer Amazon S3-Daten definieren können. Weitere Informationen finden Sie unter Verwenden von AWS Backup für Amazon S3.	30. November 2021
AWS Identity and Access Management Access Analyzer für Amazon S3	IAM Access Analyzer führt Richtlinienprüfungen durch, um Ihre Richtlinie anhand der IAM-Richtliniengrammatik und der bewährten Methoden zu validieren. Weitere Informationen zum Validieren von Richtlinien mit IAM Access Analyzer finden Sie unter Validierung der IAM-Access-Analyzer-Richtlinien im IAM-Benutzerhandbuch.	30. November 2021
Neue Ereignistypen	Zu Amazon-S3-Ereignisbenachrichtigungen hinzugefügte Ereignistypen finden Sie unter Amazon-S3-Ereignisbenachrichtigungen.	29. November 2021
Aktivieren von Amazon EventBridge in Buckets	Sie können EventBridge in Amazon S3-Buckets aktivieren, um Ereignisse an Amazon zu senden EventBridge, siehe Verwenden von EventBridge.	29. November 2021

Neue S3-Lebenszyklusfilter	Sie können Lebenszyklusregeln basierend auf der Objektgröße erstellen oder angeben, wie viele nicht aktuelle Objektversionen beibehalten werden sollen. Weitere Informationen finden Sie unter Beispiele der S3-Lebenszyklus-Konfiguration.	23. November 2021
Veröffentlichen von Amazon S3-Storage-Lens-Metriken in Amazon CloudWatch	Sie können Nutzungs- und Aktivitätsmetriken von S3 Storage Lens in Amazon CloudWatch veröffentlichen, um eine einheitliche Ansicht Ihres Betriebszustands in CloudWatch Dashboards zu erstellen. Sie können auch CloudWatch Funktionen wie Alarme und ausgelöste Aktionen, Metrikberechnungen und Anomalieerkennung verwenden, um S3-Storage-Lens-Metriken zu überwachen und Maßnahmen zu ergreifen. Darüber hinaus ermöglichen die CloudWatch APIs Anwendungen, einschließlich Drittanbietern, den Zugriff auf Ihre S3-Storage-Lens-Metriken. Weitere Informationen finden Sie unter Überwachen von S3-Storage-Lens-Metriken in CloudWatch.	22. November 2021

Multiregionale Zugriffspunkte	Sie können Multi-Region Access Points verwenden, um einen globalen Endpunkt zu erstellen, mit dem Anwendungen Anforderungen von Amazon-S3-Buckets in mehreren AWS-Regionen ausführen können. Sie können diesen Multi-Region Access Point verwenden, um Daten an einen Bucket mit der niedrigsten Latenz weiterzuleiten. Weitere Informationen zu Multi-Regions-Zugriffspunkte und deren Verwendung finden Sie unter Multi-Regions-Zugriffspunkte in Amazon S3.	2. September 2021

Amazon S3 in Outposts fügt direkten lokalen Zugriff für Anwendungen hinzu	Führen Sie Ihre Anwendungen außerhalb der AWS Outposts Virtual Private Cloud (VPC) aus und greifen Sie auf Ihre S3-on-Outposts-Daten zu. Sie können auch direkt aus Ihrem lokalen Netzwerk auf S3 in Outposts zugreifen. Weitere Informationen zum Konfiguri eren von S3-in-Outposts-End punkten mit kundeneigenen IP-Adressen (CoIP) und zum Zugreifen auf Ihre Objekte durch Erstellen eines lokalen Gateways aus Ihrem lokalen Netzwerk finden Sie unter Zugreifen auf Amazon S3 in Outposts mithilfe von Nur-VPC-Zugriffspunkten.	29. Juli 2021
Amazon-S3-Zugriffspunktalia se	Wenn Sie einen Zugriffspunkt erstellen, generiert Amazon S3 automatisch einen Alias, den Sie anstelle eines Bucket-Namens für den Datenzugr iff verwenden können. Sie können diesen Zugriffspunkt-Alias anstelle eines Amazon-Ressourcennamen (ARNs) für jeden Zugriffspunkt-Date nebenen-Vorgang verwenden . Weitere Informationen finden Sie unter Verwenden eines Alias im Bucket-Stil für Ihren Zugriffspunkt.	26. Juli 2021

Amazon S3 Inventory und S3 Batch Operations unterstützen den Status des S3-Bucket-Schlüssels	Amazon-S3-Bestand und Batch-Vorgänge unterstützen das Identifizieren und Kopieren vorhandener Objekte mit S3-Bucket-Schlüssel. S3-Bucket-Schlüssel beschleunigen die Preissenkung der serverseitigen Verschlüsselung für vorhandene Objekte. Weitere Informationen finden Sie unter Amazon S3 Inventory und Batch Operations Copy object (Batch-Vorgänge Objekt kopieren).	3. Juni 2021
Amazon S3 Storage-Lens Metriken Konto-Snapshot	Der S3-Storage-Lens-Konto-Snapshot zeigt Ihren Gesamtspeicher, die Objektzahl und die durchschnittliche Objektgröße auf der Startseite der S3-Konsole (Buckets) an, indem Metriken aus Ihrem Standard-Dashboard zusammengefasst werden. Weitere Informationen finden Sie unter S3 Storage Lens Metrics Konto-Snapshot.	5. Mai 2021
Erhöhte Unterstützung von Amazon-S3-in-Outposts-Endpunkten	S3 in Outposts unterstützt jetzt bis zu 100 Endpunkte pro Outpost. Weitere Informationen finden Sie unter S3-in-Outposts-Netzeinschränkungen.	29. April 2021

Ereignisbenachrichtigungen für Amazon S3 on Outposts in Amazon CloudWatch Events	Sie können CloudWatch Ereignisse verwenden, um eine Regel zu erstellen, um jedes S3-on-Outposts-API-Ereignis zu erfassen und über alle unterstützten CloudWatch Ziele benachrichtigt zu werden. Weitere Informationen finden Sie unter Empfangen von S3-on-Outposts-Ereignisbenachrichtigungen mit - CloudWatch Ereignissen.	19. April 2021
S3 Objekt Lambda	Mit S3 Object Lambda können Sie GET-Anforderungen von Amazon S3 eigenen Code hinzufügen, um Daten zu ändern und zu verarbeiten, wenn sie an eine Anwendung zurückgegeben werden. Sie können mit benutzerdefiniertem Code die von standardmäßigen S3-GET-Anforderungen zurückgegebenen Daten ändern, um u. a. Zeilen zu filtern, Bilder dynamisch in der Größe zu ändern und vertrauliche Daten zu redigieren und mehr. Weitere Informationen finden Sie unter Transformieren von Objekten.	18. März 2021

AWS PrivateLink	Mit AWS PrivateLink für Amazon S3 können Sie eine direkte Verbindung zu S3 herstellen, indem Sie einen Schnittstellenendpunkt in Ihrer Virtual Private Cloud (VPC) verwenden, anstatt eine Verbindung über das Internet herzustellen. Schnittstellen-Endpunkte sind direkt von Anwendungen aus zugänglich, die sich vor Ort oder in einer anderen AWS-Region befinden. Weitere Informationen finden Sie unter AWS PrivateLink für Amazon S3.	2. Februar 2021
Verwalten der Kapazität von Amazon S3 on Outposts mit AWS CloudTrail	Verwaltungsereignisse von S3 on Outposts sind über - CloudTrail Protokolle verfügbar. Weitere Informationen finden Sie unter Verwalten der Kapazität von S3 on Outposts mit CloudTrail.	21. Dezember 2020

Starke Konsistenz	Amazon S3 bietet eine starke read-after-write Konsistenz für - PUT und -DELETEAnfragen von Objekten in Ihrem S3-Bucket in allen AWS-Regionen. Darüber hinaus sind Lesevorgänge in Amazon S3 Select, Amazon-S3-Zugriffs kontrolllisten, Amazon-S3-Objekt-Tags und Objekt-Me tadaten (z. B. HEAD-Objekt) durchweg konsistent. Weitere Informationen finden Sie unter Amazon-S3-Datenkon sistenzmodell.	1. Dezember 2020
Synchronisierung der Amazon-S3-Replikatänderung	Die Synchronisierung von Amazon-S3-Replikat änderungen synchronisiert Objektmetadaten wie Tags, ACLs und Objektsperre-Einstellungen zwischen Quellobjekten und Replikate n. Wenn diese Funktion aktiviert ist, repliziert Amazon S3 Metadatenänderungen, die entweder am Quellobje kt oder an den Replikatk opien vorgenommen wurden. Weitere Informationen finden Sie unter Replizieren von Metadatenänderungen mit der Synchronisierung von Replikatänderungen.	1. Dezember 2020

Amazon-S3-Bucket-Schlüssel	Amazon-S3-Bucket-Schlüssel senken die Kosten für serverseitige Amazon-S3-Verschlüsselung mit AWS Key Management Service (SSE-KMS). Dieser neue Bucket-Level-Schlüssel für serversei tige Verschlüsselung kann die Kosten für AWS KMS -Anfr agen um bis zu 99 Prozent senken, indem der Anforderu ngsdatenverkehr von Amazon S3 zu AWS KMS verringert wird. Weitere Informationen finden Sie unter Reduzierung der Kosten für SSE-KMS mit S3-Bucket-Schlüsseln.	1. Dezember 2020

Amazon-S3-Storage-Lens	S3 Storage Lens aggregiert Ihre Metriken und zeigt die Informationen im Abschnitt Account snapshot (Konto-Snapshot) der Seite Buckets der Amazon-S3-Konsole an. S3 Storage Lens bietet außerdem ein interaktives Dashboard, mit dem Sie Erkenntnisse und Trends visualisieren, Ausreißer kennzeichnen und Empfehlungen zur Optimierung der Speicherkosten erhalten und bewährte Datenschutzmethoden anwenden können. Das Dashboard verfügt über Drilldown-Optionen, um Erkenntnisse auf Organisations-, Konto-, AWS-Region-, Speicherklassen-, Bucket-, Präfix- oder Objekt- oder Storage-Lens-Gruppenebene zu generieren. Sie können zudem täglich einen Metrikexport im CSV- oder Parquet-Format an einen S3-Bucket senden. Weitere Informationen finden Sie unter Bewertung Ihrer Speicheraktivität und -nutzung mit S3 Storage Lens.	18. November 2020

Nachverfolgen von S3-Anforderungen mit AWS X-Ray	Amazon S3 lässt sich in X-Ray integrieren, um den Trace-Kontext zu verbreiten und Ihnen eine Anforderungskette mit Upstream- und Downstream-Knoten bereitzustellen. Weitere Informationen finden Sie unter Verfolgen von Anforderungen mit X-Ray.	16. November 2020
S3-Replikationsmetriken	S3-Replikationsmetriken enthalten detaillierte Metriken für die Replikationsregeln in Ihrer Replikationskonfiguration. Weitere Informationen finden Sie unter Replikationsmetriken und Amazon-S3-Ereignis-Benachrichtigungen.	9. November 2020
S3-Intelligent-Tiering-Archivzugriff und tiefer Archivzugriff	S3-Intelligent-Tiering-Archivzugriff und tiefer Archivzugriff sind zusätzliche Speicherstufen unter S3-Intelligent-Tiering. Weitere Informationen finden Sie unter Speicherklasse zum automatischen Optimieren häufig und selten aufgerufener Objekte.	9. November 2020

Löschmarkierungsreplikation	Mit der Replikation von Löschmarkierungen können Sie sicherstellen, dass Löschmarkierungen für Ihre Replikationsregeln in Ihre Ziel-Buckets kopiert werden. Weitere Informationen finden Sie unter Verwenden der Löschmarkierungs-Replikation.	9. November 2020
S3 Object Ownership	Object Ownership ist eine S3-Bucket-Einstellung, mit der Sie die Eigentümerschaft an neuen Objekten, die in Ihre Buckets hochgeladen werden, steuern können. Weitere Informationen finden Sie unter Verwenden von S3-Objekt-Eigentümerschaft.	2. Oktober 2020

Amazon S3 in Outposts	Mit Amazon S3 on Outposts können Sie S3-Buckets auf Ihren AWS Outposts Ressourcen erstellen und Objekte für Anwendungen, die einen lokalen Datenzugriff, eine lokale Datenverarbeitung und eine lokale Datenresi denz erfordern, einfach On-Premises speichern und abrufen. Sie können S3 on Outposts über die AWS Management Console, AWS CLI, AWS SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter Verwenden von Amazon S3 in Outposts.	30. September 2020
Bedingung „Bucket-E igentümer“	Sie können die Amazon S3-Bucket-Eigentümerbedingu ng verwenden, um sicherzus tellen, dass die Buckets, die Sie in Ihren S3-Operationen verwenden AWS-Konten , zu den von Ihnen erwartete n gehören. Weitere Informati onen finden Sie unter Bucket-Eigentümer-Bedingung.	11. September 2020

Unterstützung von S3-Batch-Vorgang für die Aufbewahrung von Objektsperren	Sie können jetzt Batch-Vorgänge mit der S3-Objektsperre verwenden, um Aufbewahrungs-Einstellungen auf viele Amazon-S3-Objekte gleichzeitig anzuwenden. Weitere Informationen finden Sie unter Festlegen von Aufbewahrungsfristen für die S3-Objektsperre mit S3-Batchvorgängen.	4. Mai 2020
Unterstützung von S3-Batchoperationen für die gesetzliche Aufbewahrungspflicht von Objektsperren	Sie können jetzt Batchvorgänge mit der S3-Objektsperre verwenden, um vielen Amazon-S3-Objekten gleichzeitig eine gesetzliche Aufbewahrungsfrist zuzuweisen. Weitere Informationen finden Sie unter Verwenden von S3-Batch-Vorgänge zum Festlegen von gesetzlichlichen Aufbewahrungsfristen für die S3-Objektsperre.	4. Mai 2020
Aufgaben-Tags für S3-Batchvorgängen	Sie können Ihren S3-BatchVorgängeaufträgen Markierungen hinzufügen, um diese Aufträge zu steuern und zu kennzeichnen. Weitere Informationen finden Sie unter Tags für S3-Batchoperationeaufträge.	16. März 2020

Amazon-S3-Zugriffspunkte	Amazon-S3-Zugriffspunkte vereinfachen die skalierbare Verwaltung des Datenzugriffs auf freigegebene Datensätze in S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie S3-Objekt-Vorgänge ausführen können. Weitere Informationen finden Sie unter Verwalten des Datenzugriffs mit Amazon-S3-Zugriffspunkten.	02. Dezember 2019
Zugriffs-Analyzer für Amazon S3	Access Analyzer für Amazon S3 macht Sie auf S3-Buckets aufmerksam, die so konfiguriert sind, dass jedem im Internet oder anderen , einschließlich Konten außerhalb Ihrer Organisation AWS-Konten, Zugriff gewährt wird. Weitere Informationen finden Sie unter Verwenden des Zugriffs-Analyzer für Amazon S3.	02. Dezember 2019

S3-Replikationszeitkontrolle (S3 RTC)	S3-Replikationszeitkontrolle (S3 RTC) repliziert die meisten Objekte, die Sie zu Amazon S3 hochladen, in Sekunden, und 99,99 Prozent dieser Objekte innerhalb von 15 Minuten. Weitere Informationen finden Sie unter Replikation von Objekten mit Begrenzung der S3-Replikationszeit (S3 RTC).	20. November 2019
Same-Region Replication (SRR, Replikation in derselben Region)	Mit der Replikation innerhalb derselben Region (Same-Region Replication, SRR) können Objekte in Amazon-S3-Buckets in derselben AWS-Region kopiert werden. Informationen zur regionsübergreifenden Replikation (Cross-Region Replication, CRR) und zur Replikation in derselben Region finden Sie unter Replikation.	18. September 2019
Unterstützung der regionsübergreifenden Replikation für die S3-Objektsperre	Die regionsübergreifende Replikation unterstützt jetzt Objektsperren. Weitere Informationen finden Sie unter Was repliziert Amazon S3?.	28. Mai 2019

S3-Batch-Vorgänge	Mit S3-Batchvorgängen können Sie groß angelegte Batchvorgänge für Amazon-S3-Objekte durchführen. S3-Batch-Vorgänge kann eine einzelne Operation für Listen von Objekten ausführen, die Sie angeben. Ein einziger Auftrag kann die festgeleg te Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführ en. Weitere Informationen finden Sie unter Durchführen von S3-Batch-Vorgänge.	30. April 2019
Region Asien-Pazifik (Hongkong)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Hongkong) verfügbar. Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Refer enz.	24. April 2019
Hinzufügung eines neuen Feldes zu den Serverzug riffsprotokollen	In Amazon S3 wurde das folgende neue Feld zu den Server-Zugriffsprotokollen hinzugefügt: Transport Layer Security (TLS)-Version. Weitere Informationen finden Sie unter Server-Zugriffspro tokollformat.	28. März 2019

Neue Archivspeicherklasse	Amazon S3 bietet eine neue Archivspeicherklasse, S3 Glacier Deep Archive (`DEEP_ARCHIVE`), zum Speichern selten benötigte r Objekte. Weitere Informati onen finden Sie unter Speicherklassen.	27. März 2019
Hinzufügung neuer Felder zu den Serverzugriffsprotokollen	In Amazon S3 wurden die folgenden neuen Felder zu den Serverzugriffs-Pro tokollen hinzugefügt: Host-ID, Signature Version (Signatur e Version), Cipher Suite, Authentication Type (Authentif izierungstyp) und Host Header. Weitere Informationen finden Sie unter Server-Zu griffsprotokollformat.	5. März 2019
Unterstützung für Amazon-S3-Inventory-Dateien im Parquet-Format	Amazon S3 unterstützt jetzt das Format Apache Parquet (Parquet) zusätzlich zu den Dateiformaten Apache Optimized Row Columnar (ORC) und Comma-Sep arated Values (CSV). Weitere Informationen finden Sie unter Bestand.	4. Dezember 2018

S3-Objektsperre	Amazon S3 bietet jetzt Objektsperren-Funktionalität mit Write Once Read Many (WORM)-Schutz für Amazon-S3-Objekte. Weitere Informationen finden Sie unter Sperren von Objekten.	26. November 2018
Wiederherstellungsgeschwindigkeit-Upgrade	Mit dem Amazon-S3-Wiederherstellungs-Geschwindigkeits-Upgrade können Sie während der Ausführung einer Wiederherstellung aus der S3-Speicherklasse Glacier Flexible Retrieval die Geschwindigkeit der Wiederherstellung erhöhen. Weitere Informationen finden Sie unter Wiederherstellen archivierter Objekte.	26. November 2018
Wiederherstellen von Ereignisbenachrichtigungen	Amazon-S3-Ereignisbenachrichtigungen unterstützen jetzt Initiierungs- und Abschlussereignisse, wenn Objekte aus der S3-Speicherklasse Glacier Flexible Retrieval wiederhergestellt werden. Weitere Informationen finden Sie unter Ereignis-Benachrichtigungen.	26. November 2018

PUT direkt zur Speicherklasse S3 Glacier Flexible Retrieval	Die PUT-Operation von Amazon S3 unterstützt jetzt beim Erstellen von Objekten die Angabe von S3 Glacier Flexible Retrieval als Speicherklasse. Zuvor mussten Sie Objekte aus anderen Amazon-S3-Speicherklassen in die S3-Speicherklasse Glacier Flexible Retrieval übertragen. Beim Verwenden der regionsübergreifenden Replikation in S3 (Cross Region Replication, CRR) können Sie jetzt auch für replizierte Objekte S3 Glacier Flexible Retrieval als Speicherklasse angeben. Weitere Informationen zur Speicherklasse S3 Glacier Flexible Retrieval finden Sie unter Speicherklassen. Weitere Informationen zum Angeben der Speicherklasse für replizierte Objekte finden Sie unter Übersicht über die Replikations-Konfiguration. Weitere Informationen zu den REST-API-Änderungen für den direkten PUT zu S3 Glacier Flexible Retrieval finden Sie unter Dokumentverlauf: direkter PUT zu S3 Glacier Flexible Retrieval.	26. November 2018

Neue Speicherklasse	Amazon S3 stellt jetzt eine neue Speicherklasse mit dem Namen S3 Intelligent-Tierung (`INTELLIGENT_TIERING`) bereit, die für Langzeitdaten mit veränderlichen oder unbekannten Zugriffsmustern entwickelt wurde. Weitere Informationen finden Sie unter Speicherklassen.	26. November 2018
Amazon S3 Block Public Access	Amazon S3 bietet jetzt die Möglichkeit, den öffentlichen Zugriff auf Buckets und Objekte auf Bucket- oder Kontobasis zu blockieren. Weitere Informationen finden Sie unter Verwenden von Amazon S3 Block Public Access.	15. November 2018

Filtern von Erweiterungen in Regeln für die regionsübergreifende Replikation (CRR)	Sie können nun in einer CRR-Regelkonfiguration einen Objektfilter angeben, um eine Untermenge von Objekten auszuwählen, auf die die Regel angewendet werden soll. Zuvor konnten Sie ausschließlich nach Objektschlüsselpräfixen filtern. In dieser Version können Sie nach Objektschlüsselpräfixen, einem oder mehreren Objekt-Markierungen oder beidem filtern. Weitere Informationen finden Sie unter CRR-Einrichtung: Übersicht über die Replikations-Konfiguration.	19. September 2018
Neue Amazon S3 Select-Funktionen	Amazon S3 Select unterstützt jetzt Apache-Parquet-Eingaben, Abfragen für verschachtelte JSON-Objekte und zwei neue Amazon- CloudWatch Überwachungsmetriken (`SelectScannedBytes` und `SelectReturnedBytes`).	5. September 2018
Aktualisierungen jetzt über RSS verfügbar	Sie können jetzt einen RSS-Feed abonnieren, um Benachrichtigungen über Updates im Amazon-S3-Benutzerhandbuch zu erhalten.	19. Juni 2018

Frühere Updates

In der folgenden Tabelle sind die wichtigen Änderungen in jeder Version des Amazon-S3-Benutzerhandbuchs vor dem 19. Juni 2018 beschrieben.

Änderungen	Beschreibung	Datum
Aktualisierung von Codebeispielen	Aktualisierte Codebeispiele: • C# – Alle Beispiele wurden für die Verwendung des aufgabenbasierten asynchronen Musters aktualisiert. Weitere Informationen finden Sie unter Asynchrone Amazon-Web-Services-APIs für .NET im AWS SDK for .NET -Entwicklerhandbuch. Codebeispiele sind nun mit Version 3 des AWS SDK for .NET konform. • Java – Alle Beispiele wurden für die Verwendung des Client Builder-Modells aktualisiert. Weitere Informationen zum Client Builder-Modell finden Sie unter Erstellen von Service-Clients. • PHP – Alle Beispiele wurden für die Verwendung mit dem AWS SDK for PHP 3.0 aktualisiert. Weitere Informationen zu AWS SDK for PHP 3.0 finden Sie unter AWS SDK for PHP. • Ruby – Der Beispielcode wurde aktualisiert, sodass die Beispiele mit der AWS SDK for Ruby Version 3 funktioni eren.	30. April 2018
Amazon S3 meldet jetzt die `ONEZONE_IA` Speicherklassen S3 Glacier Flexible Retrieval und an Amazon- CloudWatc	Diese Speichermetriken melden nicht nur tatsächliche Byte-Zahlen, sondern beinhalten auch Overhead-Bytes pro Objekt für entsprechende Speicherklassen (`ONEZONE_IA` , `STANDARD_IA` und S3 Glacier Flexible Retrieval): • Für Objekte der Speicherklassen `ONEZONE_IA` und `STANDARD_IA` meldet Amazon S3 Objekte kleiner als	30. April 2018

Änderungen	Beschreibung	Datum
h Logs-Speichermetriken	128 KB als 128 KB. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen. • Für Objekte der Speicherklasse S3 Glacier Flexible Retrieval melden die Speichermetriken die folgenden Overheads: • Ein Overhead von 32 KB pro Objekt zu dem Preis, der für Speicherklasse S3 Glacier Flexible Retrieval berechnet wird • Ein Overhead von 8 KB pro Objekt zu dem Preis, der für Speicherklasse `STANDARD` berechnet wird Weitere Informationen finden Sie unter Übergang von Objekten mit Amazon-S3-Lebenszyklus. Weitere Informationen zu Speichermetriken finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.	
Neue Speicherklasse	Amazon S3 bietet jetzt eine neue Speicherklasse, `STANDARD_IA` (IA für „Infrequent Access“ (seltener Zugriff)), um Objekte zu speichern. Diese Speicherklasse ist für langlebige Daten und Daten mit weniger häufigem Zugriff optimiert. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen.	4. April 2018
Amazon S3 Select	Amazon S3 unterstützt jetzt den Abruf von Objektinhalten auf Basis eines SQL-Ausdrucks. Weitere Informationen finden Sie unter Filtern und Abrufen von Daten mit Amazon S3 Select.	4. April 2018

Änderungen	Beschreibung	Datum
Region Asien-Pazifik (Osaka – regional)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Osaka – regional) verfügbar. Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Refer enz. **Important** Sie können die Region Asien-Pazifik (Osaka – regional) nur in Verbindung mit der Region Asien-Pazifik (Tokio) verwenden. Um Zugang zur Region Asien-Pazifik (Osaka – regional) anzufordern, wenden Sie sich an Ihren Vertriebsmitarbeiter	12. Februar 2018
Erstellungszeitste mpel für Amazon S3 Inventory	Amazon S3 Inventory beinhaltet jetzt einen Zeitstempel mit dem Datum und der Uhrzeit der Erstellung des Amazon-S3-Bestandsberichts. Mithilfe des Zeitstempels können Sie Änderungen in Ihrem Amazon-S3-Speicher ab dem Startzeitpunkt, zu dem der Bestandsbericht erstellt wurde, ermitteln.	16. Januar 2018
Region Europa (Paris)	Amazon S3 ist jetzt in der Region Europa (Paris) verfügbar . Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.	18. Dezember 2017
Region „China (Ningxia)“	Amazon S3 ist jetzt in der Region China (Ningxia) verfügbar Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.	29. November 2017

Änderungen	Beschreibung	Datum
Unterstützung für ORC-formatierte Amazon-S3-Inventory-Dateien	Amazon S3 unterstützt jetzt das Apache Optimized Row Columnar (ORC)-Format zusätzlich zum Comma Separated Values (CSV)-Dateiformat für Bestandsausgabedateien. Außerdem können Sie jetzt Amazon S3 Inventory mit Standard-SQL abfragen, indem Sie Amazon Athena, Amazon Redshift Spectrum und andere Tools wie Presto, Apache Hive und Apache Spark verwenden. Weitere Informationen finden Sie unter Amazon S3 Inventory.	17. November 2017
Standard-Verschlüsselung für S3-Buckets	Die Amazon-S3-Standard-Verschlüsselung bietet eine Methode zum Festlegen des Verhaltens der Standard-Verschlüsselung für einen S3-Bucket. Sie können die Standard-Verschlüsselung in einem Bucket festlegen, sodass alle Objekte beim Speichern im Bucket verschlüsselt werden. Die Objekte werden mittels serverseitiger Verschlüsselung verschlüsselt, entweder mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) oder mit AWS von verwalteten Schlüsseln (SSE-KMS). Weitere Informationen finden Sie unter Einstellen des Verhaltens der serverseitigen Verschlüsselung für Amazon S3-Buckets.	06. November 2017
Verschlüsselungsstatus im Amazon S3 Inventory	Amazon S3 unterstützt jetzt das Einsetzen eines Verschlüsselungsstatus in Amazon S3 Inventory, sodass Sie sehen können, wie Ihre Objekte im Ruhezustand für die Überwachung der Compliance von Vorschriften oder andere Zwecke verschlüsselt werden. Sie können auch so konfigurieren, dass Amazon-S3-Bestand mit serverseitiger Verschlüsselung (SSE) oder SSE-KMS verschlüsselt wird, damit alle Bestandsdateien entsprechend verschlüsselt werden. Weitere Informationen finden Sie unter Amazon S3 Inventory.	06. November 2017

Änderungen	Beschreibung	Datum
Erweiterungen der regionsübergreifenden Replikation (CRR)	Die regionsübergreifende Replikation unterstützt jetzt Folgendes: • In einem kontoübergreifenden Szenario können Sie eine CRR-Konfiguration hinzufügen, um die Replikat-Eigentümerschaft dem AWS-Konto zuzuweisen, das den Ziel-Bucket besitzt. Weitere Informationen finden Sie unter Ändern des Replikat-Eigentümers. • Standardmäßig repliziert Amazon S3 keine Objekte in Ihrem Quell-Bucket, die mit serverseitiger Verschlüsselung mit in gespeicherten Schlüsseln erstellt wurden AWS KMS . In Ihrer CRR-Konfiguration können Sie jetzt Amazon S3 anweisen, diese Objekte zu replizieren. Weitere Informationen finden Sie unter Replizieren von mit serverseitiger Verschlüsselung (SSE-C, SSE-S3, SSE-KMS, DSSE-KMS) erstellten Objekten.	06. November 2017
Region Europa (London)	Amazon S3 ist jetzt in der Region Europa (London) verfügbar. Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.	13. Dezember 2016
Region Kanada (Zentral)	Amazon S3 jetzt in der Region Kanada (Zentral) verfügbar Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.	8. Dezember 2016

Änderungen	Beschreibung	Datum
Markieren von Objekten	Amazon S3 unterstützt jetzt das Markieren von Objekten. Das Markieren von Objekten ermöglicht Ihnen, Speicher zu kategorisieren. Präfixe für Objektschlüsselnamen ermöglichen Ihnen außerdem, die Speicherung zu kategorisieren. Die Markierung von Objekten fügt ihr eine weitere Dimension hinzu. Die Markierung bietet weitere Vorteile. Dazu zählen: • Objekt-Markierungen bieten eine differenzierte Zugriffskontrolle für Berechtigungen (Sie könnten beispielsweise einem IAM-Benutzer die Berechtigungen für schreibgeschützte Objekte mit bestimmten Markierungen erteilen). • Differenzierte Kontrolle bei der Spezifizierung einer Lebenszyklus-Konfiguration. Sie können Markierungen spezifizieren, um eine Untermenge von Objekten auszuwählen, auf welche die Lebenszyklus-Regel zutrifft. • Wenn Sie eine regionsübergreifende Replikation (CRR) konfiguriert haben, kann Amazon S3 die Markierungen replizieren. Sie müssen der für Amazon S3 erstellten IAM-Rolle die erforderlichen Berechtigungen erteilen, um Objekte zu replizieren. • Sie können auch CloudWatch Metriken und CloudTrail Ereignisse anpassen, um Informationen nach bestimmten Tag-Filtern anzuzeigen. Weitere Informationen finden Sie unter Kategorisieren des Speichers mithilfe von Markierungen.	29. November 2016

Änderungen	Beschreibung	Datum
Amazon S3 Lifecycle unterstützt jetzt tagbasierte Filter	Amazon S3 unterstützt jetzt auf Tags basierende Filter bei der Konfiguration des Lebenszyklus. Sie können jetzt Lebenszyklusregeln angeben, in denen Sie ein Schlüsselpräfix, ein oder mehrere Objekt-Markierungen oder eine Kombination aus beidem angeben, um eine Untermenge der Objekte auszuwählen, auf welche die Lebenszyklusregel zutrifft. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.	29. November 2016
CloudWatch Anforderungsmetriken für Buckets	Amazon S3 unterstützt jetzt CloudWatch Metriken für Anforderungen an Buckets. Wenn Sie diese Metriken für einen Bucket aktivieren, erstellen diese Metriken Berichte in 1-Minuten-Intervallen. Sie können auch konfigurieren, welche Objekte in einem Bucket diese Anforderungsmetriken melden. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.	29. November 2016
Amazon S3 Inventory	Amazon S3 unterstützt jetzt den Speicherbestand. Amazon S3 Inventory stellt eine Flatfile-Dateiausgabe mit durch Kommata getrennten Werten (.csv) Ihrer Objekte und der zugehörigen Metadaten auf täglicher oder wöchentlicher Basis für einen S3-Bucket oder ein gemeinsames Präfix (d. h. Objekte, deren Namen mit derselben Zeichenfolge beginnen) bereit. Weitere Informationen finden Sie unter Amazon S3 Inventory.	29. November 2016

Änderungen	Beschreibung	Datum
Amazon S3 Analytics – Speicherklassen-Analyse	Die neue analytische Funktion von Amazon S3, die Speicherklassen-Analyse, beobachtet Datenzugriffsmuster, anhand derer Sie entscheiden können, wann Sie `STANDARD`-Speicher mit weniger häufigem Zugriff in die Speicherklasse `STANDARD_IA` (IA für „infrequent access“ (seltener Zugriff)) überführen sollen. Wenn die Speicherklassen-Analyse die seltenen Zugriffsmuster für eine gefilterte Datenmenge im Laufe der Zeit beobachtet, können Sie Ihre Lebenszykluskonfigurationen unter Verwendung der Analyseergebnisse verbessern. Diese Funktion beinhaltet auch eine detaillierte tägliche Analyse Ihrer Speichernutzung auf der angegebenen Bucket-, Präfix- oder Tag-Ebene, die Sie in einen S3-Bucket exportieren können.	29. November 2016
Neuer Expedited- und Bulk-Datenabrufe bei der Wiederherstellung archivierter Objekte aus S3 Glacier	Amazon S3 unterstützt jetzt den Expedited- und Bulk-Datenabruf zusätzlich zum Standardabruf bei der Wiederherstellung archivierter Objekte in S3 Glacier. Weitere Informationen finden Sie unter Wiederherstellen eines archivierten Objekts.	21. November 2016
CloudTrail -Objektprotokollierung	CloudTrail unterstützt die Protokollierung von Amazon S3-API-Operationen auf Objektebene wie `GetObject` `PutObject` , und `DeleteObject` . Sie können Ihre Ereignisauswahlen so konfigurieren, dass API-Vorgänge auf Objektebene protokolliert werden. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail.	21. November 2016
Region USA Ost (Ohio)	Amazon S3 ist jetzt in der Region USA Ost (Ohio) verfügbar. Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.	17. Oktober 2016

Änderungen	Beschreibung	Datum
IPv6-Unterstützung für Amazon S3 Transfer Acceleration	Amazon S3 unterstützt jetzt Internet Protocol Version 6 (IPv6) für Amazon S3 Transfer Acceleration. Sie können eine Verbindung zu Amazon S3 über IPv6 herstellen, indem Sie den neuen Dual-Stack for Transfer Acceleration-Endpunkt verwenden. Weitere Informationen finden Sie unter Erste Schritte mit Amazon S3 Transfer Acceleration.	6. Oktober 2016
IPv6-Support	Amazon S3 unterstützt jetzt Internet Protocol Version 6 (IPv6). Sie können über IPv6; unter Verwendung von Dual-Stack-Endpunkten auf Amazon S3 zugreifen. Weitere Informationen finden Sie unter Stellen von Anforderungen an Amazon S3 über IPv6.	11. August 2016
Region Asien-Pazifik (Mumbai)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Mumbai) verfügbar. Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.	27. Juni 2016
Amazon S3 Transfer Acceleration	Amazon S3 Transfer Acceleration ermöglicht die schnelle, einfache und sichere Übertragung von Dateien über große Entfernungen zwischen Ihrem Client und einem S3-Bucket. Transfer Acceleration nutzt CloudFront die global verteilten Edge-Standorte von Amazon. Weitere Informationen finden Sie unter Konfigurieren schneller, sicherer Dateiübertragungen mit Amazon S3 Transfer Acceleration.	19. April 2016
Lebenszyklusunterstützung, um abgelaufene Löschmarkierungen für Objekte zu entfernen	Die `Expiration` -Aktion der Lebenszykluskonfiguration erlaubt jetzt, Amazon S3 anzuweisen, abgelaufene Löschmarkierungen für Objekte in einem versionsgesteuerten Bucket zu entfernen. Weitere Informationen finden Sie unter Elemente, die Lebenszyklus-Aktionen beschreiben.	16. März 2016

Änderungen	Beschreibung	Datum
Die Bucket-Lebenszyklus-Konfiguration unterstützt jetzt eine Aktion, um unvollständige mehrteilige Uploads abzubrechen	Die Bucket-Lebenszykluskonfiguration unterstützt jetzt die `AbortIncompleteMultipartUpload` -Aktion. Mit dieser Aktion können Sie Amazon S3 zum Abbrechen mehrteiliger Uploads anweisen, die nicht innerhalb einer bestimmten Anzahl von Tagen nach der Initiierung abgeschlossen werden. Wenn ein mehrteiliger Upload für eine Abbruchoperation in Frage kommt, löscht Amazon S3 alle hochgeladenen Teile und bricht den mehrteiligen Upload ab. Konzeptuelle Informationen finden Sie in folgenden Themen im Amazon-S3-Benutzerhandbuch: • Abbrechen eines mehrteiligen Uploads • Elemente, die Lebenszyklus-Aktionen beschreiben Die folgenden API-Vorgänge wurden aktualisiert, um die neue Aktion zu unterstützen: • PUT Bucket lifecycle (PUT Bucket-Lebenszyklus) – Die XML-Konfiguration ermöglicht Ihnen jetzt die Angabe der `AbortIncompleteMultipartUpload` -Aktion in einer Lebenszyklus-Konfigurationsregel. • List Parts (Teile auflisten) und Initiate Multipart Upload (Mehrteiligen Upload initiieren) – Beide API-Vorgänge geben nun zwei zusätzliche Antwort-Header zurück (`x-amz-abort-date` und `x-amz-abort-rule-id`), wenn der Bucket eine Lebenszyklusregel besitzt, die die `AbortIncompleteMultipartUpload` -Aktion angibt. Diese Header in der Antwort geben an, wann der initiierte mehrteilige Upload für eine Abbruchoperation in Frage kommt, und welche Lebenszyklusregel gilt.	16. März 2016

Änderungen	Beschreibung	Datum
Region Asien-Pazifik (Seoul)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Seoul) verfügbar. Weitere Informationen zu Regionen und Endpunkten von Amazon S3 finden Sie unter Regionen und Endpunkte in der Allgemeine AWS-Referenz.	6. Januar 2016
Neuer Bedingungsschlüssel und eine Änderung des mehrteiligen Uploads	IAM-Richtlinien unterstützen jetzt einen Amazon-S3-`s3:x-amz-storage-class` -Bedingungsschlüssel. Weitere Informationen finden Sie unter Beispiele für Amazon-S3-Bedingungsschlüssel. Sie müssen nicht mehr der Initiator eines mehrteiligen Uploads sein, um Teile hochzuladen und den Upload abzuschließen. Weitere Informationen finden Sie unter API für mehrteilige Uploads und Berechtigungen.	14. Dezember 2015
Umbenennung der US-Standardregion	Die Regionsnamen-Zeichenfolge hat sich von "USA Standard" in "USA Ost (Nord-Virginia)" geändert. Dies ist nur eine Aktualisierung des Regionsnamens, die Funktionalität bleibt unverändert.	11. Dezember 2015

Änderungen	Beschreibung	Datum
Neue Speicherklasse	Amazon S3 bietet jetzt eine neue Speicherklasse, STANDARD_IA (IA für „infrequent access“ (seltener Zugriff)), um Objekte zu speichern. Diese Speicherklasse ist für langlebige Daten und Daten mit weniger häufigem Zugriff optimiert. Weitere Informationen finden Sie unter Verwenden von Amazon-S3-Speicherklassen. Die Aktualisierungen der Lebenszyklus-Konfiguration ermöglichen Ihnen jetzt, Objekte in die Speicherklasse STANDARD_IA zu überführen. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus. Zuvor hat die Funktion der regionsübergreifenden Replikation die Speicherklasse des Quellobjekts für Objektreplikate verwendet. Wenn Sie jetzt eine regionsübergreifende Replikation konfigurieren, können Sie eine Speicherklasse für das im Ziel-Bucket erstellte Objektreplikat spezifizieren. Weitere Informationen finden Sie unter Replizieren von Objekten.	16. September 2015
AWS CloudTrail - Integration	Mit der neuen AWS CloudTrail Integration können Sie Amazon S3-API-Aktivitäten in Ihrem S3-Bucket aufzeichnen. Sie können verwenden CloudTrail , um das Erstellen oder Löschen von S3-Buckets, Änderungen der Zugriffskontrolle oder Änderungen der Lebenszykluskonfiguration nachzuverfolgen. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mit AWS CloudTrail.	1. September 2015

Änderungen	Beschreibung	Datum
Erhöhung des Bucket-Limits	Amazon S3 unterstützt jetzt die Erhöhung des Bucket-Limits. Standardmäßig können Kunden bis zu 100 Buckets in ihrem erstellen AWS-Konto. Kunden, die weitere Buckets benötigen, können dieses Limit erhöhen, indem sie eine Service-Limit-Erhöhung senden. Weitere Informationen zum Erhöhen des Bucket-Kontingents finden Sie unter AWS-Service -Service-Quotas in der allgemeinen AWS -Referenz. Weitere Informationen finden Sie unter Verwenden der AWS SDKs und Beschränkungen und Einschränkungen von Buckets.	4. August 2015
Aktualisierung des Datenkonsistenzmodells	Amazon S3 unterstützt jetzt die read-after-write Konsistenz für neue Objekte, die Amazon S3 in der Region USA Ost (Nord-Virginia) hinzugefügt wurden. Vor diesem Update unterstützten alle Regionen außer USA Ost (Nord-Virginia) die read-after-write Konsistenz für neue Objekte, die auf Amazon S3 hochgeladen wurden. Mit dieser Erweiterung unterstützt Amazon S3 jetzt die read-after-write Konsistenz in allen Regionen für neue Objekte, die zu Amazon S3 hinzugefügt werden. Mit Read-after-write-Konsistenz können Sie Objekte sofort nach der Erstellung in Amazon S3 abrufen. Weitere Informationen finden Sie unter Regionen.	4. August 2015
Ereignis-Benachrichtigungen	Amazon-S3-Ereignisbenachrichtigungen wurden aktualisiert, um Benachrichtigungen hinzuzufügen, wenn Objekte gelöscht wurden, und das Filtern für Objektnamen mit Präfix- und Suffix-Vergleich hinzuzufügen. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.	28. Juli 2015

Änderungen	Beschreibung	Datum
Amazon- CloudWatch Integration	Mit der neuen Amazon- CloudWatch Integration können Sie Alarme für Ihre Amazon S3-Nutzung über CloudWatch Metriken für Amazon S3 überwachen und einstellen. Unterstützte Metriken sind unter anderem die Gesamtzahl der Bytes für den Standardspeicher, die Gesamtzahl der Bytes für den Speicher mit reduzierter Redundanz sowie die Gesamtzahl der Objekte für einen bestimmten S3-Bucket. Weitere Informationen finden Sie unter Überwachen von Metriken mit Amazon CloudWatch.	28. Juli 2015
Unterstützung für das Löschen und Leeren nicht leerer Buckets	Amazon S3 unterstützt jetzt das Löschen und Leeren nicht leerer Buckets. Weitere Informationen finden Sie unter Leeren eines Buckets.	16. Juli 2015
Bucket-Richtlinien für Amazon VPC-Endpunkte	Amazon S3 hat Unterstützung für Bucket-Richtlinien für Virtual Private Cloud (VPC)-Endpunkte hinzugefügt. Sie können den Zugriff auf S3-Buckets von bestimmten VPC-Endpunkten oder bestimmten VPCs über Bucket-Richtlinien steuern. VPC-Endpunkte sind einfach zu konfigurieren, höchst zuverlässig und bieten eine sichere Verbindung zu Amazon S3, ohne dass ein Gateway oder eine NAT-Instance erforderlich sind. Weitere Informationen finden Sie unter Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien.	29. April 2015
Ereignis-Benachrichtigungen	Amazon S3-Ereignisbenachrichtigungen wurden aktualisiert, um den Wechsel zu ressourcenbasierten Berechtigungen für - AWS Lambda Funktionen zu unterstützen. Weitere Informationen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.	9. April 2015

Änderungen	Beschreibung	Datum
Regionsübergreifende Replikation	Neu: Amazon S3 unterstützt jetzt regionsübergreifende Replikation Die regionsübergreifende Replikation ist das automatische, asynchrone Kopieren von Objekten über Buckets hinweg in verschiedenen AWS-Regionen. Weitere Informationen finden Sie unter Replizieren von Objekten.	24. März 2015
Ereignis-Benachrichtigungen	Amazon S3 unterstützt jetzt neue Ereignistypen und Ziele in einer Bucket-Benachrichtigungskonfiguration. Vor dieser Version unterstützte Amazon S3 nur den Ereignistyp s3:ReducedRedundancyLostObject und ein Amazon SNS-Thema als Ziel. Weitere Informationen zu den neuen Ereignistypen finden Sie unter Amazon-S3-Ereignis-Benachrichtigungen.	13. November 2014
Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln	Serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Schlüsseln (SSE-KMS) Amazon S3 unterstützt jetzt die serverseitige Verschlüsselung mit AWS KMS. Mit dieser Funktion können Sie den Envelope-Schlüssel über verwalten. Amazon S3 ruft auf AWS KMS, AWS KMS um innerhalb der von Ihnen festgelegten Berechtigungen auf den Envelope-Schlüssel zuzugreifen. Weitere Informationen zur serverseitigen Verschlüsselung mit AWS KMS finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung mit AWS Key Management Service.	12. November 2014
Region Europa (Frankfurt)	Amazon S3 ist jetzt in der Region Europa (Frankfurt) verfügbar.	23. Oktober 2014

Änderungen	Beschreibung	Datum
Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln	Amazon S3 unterstützt jetzt die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C). Die serverseitige Verschlüsselung ermöglicht Ihnen, Amazon S3 aufzufordern, Ihre ruhenden Daten zu verschlüsseln. Wenn Sie SSE-C verwenden, verschlüsselt Amazon S3 Ihre Objekte mit den von Ihnen bereitgestellten benutzerdefinierten Verschlüsselungsschlüsseln. Amazon S3 übernimmt die Verschlüsselung für Sie, deshalb können Sie von der Verwendung Ihrer eigenen Verschlüsselungsschlüssel profitieren, ohne die Kosten für das Schreiben oder Ausführen Ihres eigenen Verschlüsselungscodes tragen zu müssen. Weitere Informationen zu SSE-C finden Sie unter Serverseitige Verschlüsselung (Verwenden von kundenseitig bereitgestellten Verschlüsselungsschlüsseln).	12. Juni 2014
Lebenszyklus-Unterstützung für das Versioning	Vor diesem Release wurde die Lebenszyklus-Konfiguration nur für nicht versionsfähige Buckets unterstützt. Jetzt können Sie den Lebenszyklus sowohl für nicht versioning-fähige, als auch für versioning-fähige Buckets konfigurieren. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.	20. Mai 2014
Überarbeitung der Themen zur Zugriffskontrolle	Überarbeitung der Dokumentation zur Amazon-S3-Zugriffskontrolle. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3.	15. April 2014
Überarbeitung des Themas zur Protokollierung des Server-Zugriffs	Überarbeitung der Dokumentation zur Protokollierung des Server-Zugriffs. Weitere Informationen finden Sie unter Protokollieren von Anfragen mit Server-Zugriffsprotokollierung.	26. November 2013
Aktualisierung der .NET SDK-Beispiele auf Version 2.0	Die .NET SDK-Beispiele in diesem Handbuch sind jetzt konform zu Version 2.0.	26. November 2013

Änderungen	Beschreibung	Datum
SOAP-Unterstützung über HTTP veraltet	Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon-S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen Ihnen, entweder die REST-API oder die - AWS SDKs zu verwenden.	20. September 2013
Unterstützung für IAM-Richtlinienvariablen	Die IAM-Richtliniensprache unterstützt jetzt Variablen . Wenn eine Richtlinie ausgewertet wird, werden alle Richtlinienvariablen durch Werte ersetzt, die von Kontext-basierten Informationen aus der Sitzung des authentif izierten Benutzers bereitgestellt werden. Sie können Richtlinienvariablen zum Definieren von allgemeinen Richtlinien nutzen, ohne explizit alle Komponenten der Richtlinien aufzulisten. Weitere Informationen über Richtlini envariablen finden Sie unter Übersicht über IAM-Richt linienvariablen im IAM-Benutzerhandbuch. Beispiele für Richtlinienvariablen in Amazon S3 finden Sie unter Beispiele für Benutzer- und Rollenrichtlinien.	3. April 2013
Konsolensupport für die Zahlung durch den Anforderer	Sie können jetzt Ihren Bucket für Requester Pays unter Verwendung der Amazon-S3-Konsole konfigurieren. Weitere Informationen finden Sie unter Nutzen von Buckets mit Zahlung durch den Anforderer für Speicherübertragun gen und Nutzung.	31. Dezember 2012

Änderungen	Beschreibung	Datum
Root-Domänen-Unterstützung für das Website-Hosting	Amazon S3 unterstützt jetzt das Hosting statischer Websites in der Root-Domäne. Besucher Ihrer Website können jetzt von ihrem Browser aus auf Ihre Site zugreifen , ohne www in der Webadresse angeben zu müssen (sie können z. B. example.com statt www.example.com verwenden). Viele Kunden hosten bereits statische Websites auf Amazon S3, die von einer www-Subdomän e aus zugänglich sind (z. B. www.example.com). Zuvor mussten Sie für eine Unterstützung des Root-Domänen-Zugriffs Ihren eigenen Webserver betreiben, um die Root-Domänen-Anfragen über einen Proxy von den Browsern auf Ihre Website auf Amazon S3 weiterzugeben. Der Betrieb eines Web-Servers, um Anforderungen über einen Proxy weiterzugeben, führt zusätzliche Kosten, Betriebsa ufwand und einen weitere potenziellen Ausfallpunkt ein. Jetzt können Sie den Vorteil der hohen Verfügbarkeit und Robustheit von Amazon S3 für www- und Root-Domänen-Adressen nutzen. Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.	27. Dezember 2012
Überarbeitung der Konsole	Die Amazon-S3-Konsole wurde aktualisiert. Die Dokumentationsthemen, die auf die Konsole verweisen, wurden entsprechend überarbeitet.	14. Dezember 2012

Änderungen	Beschreibung	Datum
Unterstützung für die Archivierung von Daten in S3 Glacier	Amazon S3 unterstützt jetzt eine Speicheroption, die Ihnen gestattet, den kostengünstigen Speicherservice von S3 Glacier für die Datenarchivierung zu nutzen. Für die Archivierung von Objekten definieren Sie Archivierungsregeln, die Objekte identifizieren, sowie einen Zeitrahmen, der festlegt, wann Amazon S3 diese Objekte in S3 Glacier archivieren soll. Sie können die Regeln für einen Bucket einfach über die Amazon S3-Konsole oder programmgesteuert über die Amazon S3-API oder AWS SDKs festlegen. Weitere Informationen finden Sie unter Verwalten Ihres Speicher-Lebenszyklus.	13. November 2012
Unterstützung für Seitenumleitungen von Websites	Für einen Bucket, der als Website konfiguriert ist, unterstützt Amazon S3 jetzt die Umleitung einer Anfrage für ein Objekt auf ein anderes Objekt im selben Bucket oder zu einer externen URL. Weitere Informationen finden Sie unter (Optional) Konfigurieren einer Webseitenumleitung. Weitere Informationen zum Hosting von Websites finden Sie unter Hosten einer statischen Website mit Amazon S3.	4. Oktober 2012
Unterstützung von Cross-Origin Resource Sharing (CORS)	Amazon S3 unterstützt jetzt Cross-Origin Resource Sharing (CORS). CORS bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain. Mit CORS-Unterstützung in Amazon S3 können Sie umfassende clientseitige Webanwendungen mit Amazon S3 erstellen und selektiven domänenübergreifenden Zugriff auf Ihre Amazon-S3-Ressourcen zulassen. Weitere Informationen finden Sie unter Cross-Origin Resource Sharing (CORS) verwenden.	31. August 2012

Änderungen	Beschreibung	Datum
Unterstützung von Kostenzuordnungs-Markierungen	Amazon S3 unterstützt jetzt Kostenzuordnungs-Markierungen, um S3-Buckets zu markieren, sodass Sie die Kosten für Projekte oder andere Kriterien einfacher nachverfolgen können. Weitere Informationen zur Verwendung der Markierung von Buckets finden Sie unter Verwenden von Kostenzuordnungs-Markierungen für S3-Buckets.	21. August 2012
Unterstützung für MFA-geschütztem API-Zugriff in Bucket-Richtlinien	Amazon S3 unterstützt jetzt MFA-geschützten API-Zugriff, eine Funktion, die eine AWS Multi-Faktor-Authentifizierung für eine zusätzliche Sicherheitsebene beim Zugriff auf Ihre Amazon S3-Ressourcen erzwingen kann. Es handelt sich um eine Sicherheitsfunktion, die die Angabe eines gültigen MFA-Codes von Benutzern erfordert, mit dem das physische Eigentum eines MFA-Geräts belegt wird. Weitere Informationen finden Sie unter AWS -Multi-Faktor-Authentifizierung. Sie können jetzt die MFA-Authentifizierung für alle Anfragen erfordern, die auf Ihre Amazon-S3-Ressourcen zugreifen. Um die MFA-Authentifizierung zu erzwingen, unterstützt Amazon S3 jetzt den `aws:MultiFactorAuthAge` -Schlüssel in einer Bucket-Richtlinie. Ein Beispiel für eine Bucket-Richtlinie finden Sie in Beispiel Verlangen von MFA.	10. Juli 2012
Unterstützung des Ablaufens von Objekten	Sie können das Ablaufen von Objekten nutzen, um das automatische Entfernen von Daten nach einem konfigurierten Zeitraum einzuplanen. Sie legen die Ablaufzeit eines Objekts fest, indem Sie einem Bucket eine Lebenszyklus-Konfiguration hinzufügen.	27. Dezember 2011
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region Südamerika (São Paulo). Weitere Informationen finden Sie unter Zugreifen auf einen Amazon-S3-Bucket und Auflisten des Buckets.	14. Dezember 2011

Änderungen	Beschreibung	Datum
Multi-Object Delete	Amazon S3 unterstützt jetzt die Multi-Object Delete-API, mit der Sie mehrere Objekte innerhalb einer einzigen Anfrage löschen können. Mit dieser Funktion können Sie eine große Anzahl an Objekten aus Amazon S3 schneller als unter Verwendung mehrerer einzelner DELETE-Anfragen löschen. Weitere Informationen finden Sie unter Löschen von Amazon-S3-Objekten.	7. Dezember 2011
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region USA West (Oregon). Weitere Informationen finden Sie unter Buckets und Regionen.	8. November 2011
Aktualisierung der Dokumentation	Korrektur von Fehlern in der Dokumentation	8. November 2011
Aktualisierung der Dokumentation	Neben den Korrekturen von Fehlern in der Dokumentation enthält diese Version auch die folgenden Erweiterungen: • Neue serverseitige Verschlüsselungsabschnitte mit der AWS SDK for PHP und der AWS SDK for Ruby (siehe Angeben serverseitiger Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)). • Neuer Abschnitt zum Erstellen und Testen von Ruby-Beispielen (siehe Verwenden von AWS SDK for Ruby – Version 3).	17. Oktober 2011

Änderungen	Beschreibung	Datum
Unterstützung der serverseitigen Verschlüsselung	Amazon S3 unterstützt jetzt die serverseitige Verschlüsselung Diese ermöglicht Ihnen, Amazon S3 aufzufordern, Ihre ruhenden Daten zu verschlüsseln, d. h. Ihre Objektdaten zu verschlüsseln, wenn Amazon S3 Ihre Daten in seinen Rechenzentren auf Datenträger schreibt. Zusätzlich zu REST-API-Updates bieten die AWS SDK for Java und .NET die erforderlichen Funktionen, um eine serverseitige Verschlüsselung anzufordern. Sie können auch eine serverseitige Verschlüsselung anfordern, wenn Sie Objekte mit der AWS Management Console hochladen. Weitere Informationen zur Datenverschlüsselung finden Sie unter Verwenden der Datenverschlüsselung.	4. Oktober 2011
Aktualisierung der Dokumentation	Neben den Korrekturen von Fehlern in der Dokumentation enthält diese Version auch die folgenden Erweiterungen: • Dem Abschnitt Senden von Anforderungen wurden Ruby- und PHP-Beispiele hinzugefügt. • Hinzufügung von Abschnitten, die Generierung und Verwendung vorsignierter URLs beschreiben. Weitere Informationen finden Sie unter Gemeinsame Nutzung von Objekten mit vorsignierten URLs und Gemeinsame Nutzung von Objekten mit vorsignierten URLs. • Ein vorhandener Abschnitt wurde aktualisiert, um - AWS Explorer für Eclipse und Visual Studio einzuführen. Weitere Informationen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.	22. September 2011

Änderungen	Beschreibung	Datum
Unterstützung von Anforderungen unter Verwendung temporärer Sicherheitsanmeldeinformationen	Zusätzlich zur Verwendung Ihrer Sicherheitsanmeldeinformationen für AWS-Konto und IAM-Benutzer zum Senden authentifizierter Anfragen an Amazon S3 können Sie jetzt Anfragen mit temporären Sicherheitsanmeldeinformationen senden, die Sie von AWS Identity and Access Management (IAM) erhalten. Sie können die AWS Security Token Service -API oder die AWS -SDK-Wrapper-Bibliotheken verwenden, um diese temporären Anmeldeinformationen von IAM anzufordern. Sie können diese temporären Anmeldeinformationen für Ihre eigene Verwendung erhalten, oder sie verbundenen Benutzern und Anwendungen erteilen. Mit dieser Funktion können Sie Ihre Benutzer außerhalb von verwalten AWS und ihnen temporäre Sicherheitsanmeldeinformationen für den Zugriff auf Ihre AWS -Ressourcen bereitstellen. Weitere Informationen finden Sie unter Senden von Anforderungen. Weitere Informationen zur Unterstützung temporärer Anmeldeinformationen in IAM finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch.	3. August 2011

Änderungen	Beschreibung	Datum
Die API für mehrteilige Uploads wurde erweitert, um das Kopieren von Objekten bis zu 5 TB zu ermöglichen	Vor dieser Version hat die Amazon-S3-API nur das Kopieren von Objekten von bis zu 5 GB unterstützt. Um das Kopieren von Objekten mit mehr als 5 GB zu unterstützen, erweitert Amazon S3 jetzt die API für den mehrteiligen Upload mit einer neuen Operation, `Upload Part (Copy)`. Sie können diese Operation für einen mehrteiligen Upload verwenden, um Objekte mit einer Größe von 5 TB zu kopieren. Weitere Informationen finden Sie unter Objekte kopieren. Weitere konzeptuelle Informationen über mehrteilige Uploads finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.	21. Juni 2011
SOAP API-Anrufe über HTTP deaktiviert	Um die Sicherheit zu erhöhen, wurden SOAP API-Aufrufe über HTTP deaktiviert. Authentifizierte und anonyme SOAP-Anfragen müssen mit SSL an Amazon S3 gesendet werden.	6. Juni 2011
IAM ermöglicht die kontoübergreifende Delegierung	Um auf eine Amazon S3-Ressource zuzugreifen, benötigte ein IAM-Benutzer zuvor Berechtigungen sowohl vom übergeordneten als auch vom Eigentümer AWS-Konto der Amazon S3-Ressource. Mit dem kontoübergreifenden Zugriff braucht der IAM-Benutzer jetzt nur die Berechtigung vom Kontoeigentümer. Das heißt, wenn ein Ressourcenbesitzer Zugriff auf ein gewährt AWS-Konto, AWS-Konto kann das seinen IAM-Benutzern jetzt Zugriff auf diese Ressourcen gewähren. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer im IAM-Benutzerhandbuch. Weitere Informationen über die Angabe von Prinzipalen in einer Bucket-Richtlinie finden Sie unter Prinzipale.	6. Juni 2011

Änderungen	Beschreibung	Datum
Neuer Link	Die Endpunktinformationen dieses Service befinden sich jetzt in der allgemeinen AWS -Referenz. Weitere Informationen finden Sie unter Regionen und Endpunkte in der allgemeinen AWS -Referenz.	1. März 2011
Unterstützung des Hostings statischer Websites in Amazon S3	Amazon S3 führt die erweiterte Unterstützung des Hostings statischer Websites ein. Dies beinhaltet die Unterstützung von Indexdokumenten und eines benutzerdefinierten Fehlerdokuments. Wenn Sie diese Funktionen verwenden, geben Anforderungen an das Stammverzeichnis Ihres Buckets oder eines Unterordners (z. B. `http://mywebsite.com/subfolder`) Ihr Indexdokument statt der Liste der Objekte in Ihrem Bucket zurück. Wenn ein Fehler festgestellt wird, gibt Amazon S3 Ihre benutzerdefinierte Fehlermeldung statt einer Amazon-S3-Fehlermeldung zurück. Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.	6. Juni 2011
Die Endpunktinformationen dieses Service befinden sich jetzt in der allgemeinen AWS -Referenz. Weitere Informationen finden Sie unter Regionen und Endpunkte in der allgemeinen AWS -Referenz.	1. März 2011	

Änderungen	Beschreibung	Datum
Unterstützung des Hostings statischer Websites in Amazon S3	Amazon S3 führt die erweiterte Unterstützung des Hostings statischer Websites ein. Dies beinhaltet die Unterstützung von Indexdokumenten und eines benutzerdefinierten Fehlerdokuments. Wenn Sie diese Funktionen verwenden, geben Anforderungen an das Stammverzeichnis Ihres Buckets oder eines Unterordners (z. B. `http://mywebsite.com/subfolder`) Ihr Indexdokument statt der Liste der Objekte in Ihrem Bucket zurück. Wenn ein Fehler festgestellt wird, gibt Amazon S3 Ihre benutzerdefinierte Fehlermeldung statt einer Amazon-S3-Fehlermeldung zurück. Weitere Informationen finden Sie unter Hosten einer statischen Website mit Amazon S3.	17. Februar 2011
Unterstützung der Response Header API	Die GET Object REST-API unterstützt jetzt die Änderung der Antwort-Header der REST GET Object-Anforderung für jede Anforderung. Das bedeutet, Sie können Objekt-Metadaten in der Antwort ändern, ohne das eigentliche Objekt zu ändern. Weitere Informationen finden Sie unter Herunterladen von Objekten.	14. Januar 2011
Unterstützung für große Objekte	In Amazon S3 wurde die maximale Größe für ein Objekt, das Sie in einem S3-Bucket speichern können, von 5 GB auf 5 TB erhöht. Wenn Sie die REST-API verwenden, können Sie Objekte bis zu 5 GB in einer einzigen PUT-Operation hochladen. Für größere Objekte müssen Sie Multipart Upload REST-API zum Hochladen von Objekten in einzelnen Teilen verwenden. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.	9. Dezember 2010
Mehrteiliger Upload	Der mehrteilige Upload unterstützt schnellere, flexiblere Uploads in Amazon S3. Mithilfe dieser Funktion können Sie ein einzelnes Objekt in mehreren Teilen hochladen. Weitere Informationen finden Sie unter Hochladen und Kopieren von Objekten mit mehrteiligen Uploads.	10. November 2010

Änderungen	Beschreibung	Datum
Unterstützung kanonischer IDs in Bucket-Richtlinien	Sie können jetzt kanonische IDs in Bucket-Richtlinien angeben. Weitere Informationen finden Sie unter Bucket-Richtlinien und Benutzerrichtlinien.	17. September 2010
Amazon S3 arbeitet mit IAM	Dieser Service ist jetzt in AWS Identity and Access Management (IAM) integriert. Weitere Informationen finden Sie unter AWS-Services -Services, die mit IAM funktionieren im IAM-Benutzerhandbuch.	2. September 2010
Benachrichtigungen	Mit der Amazon-S3-Benachrichtigungsfunktion können Sie einen Bucket so konfigurieren, dass Amazon S3 eine Nachricht an ein Thema des Amazon Simple Notification Service (Amazon SNS) veröffentlicht, wenn Amazon S3 ein Schlüsselereignis in einem Bucket erkennt. Weitere Informationen finden Sie unter Einrichten von Benachrichtigungen für Bucket-Ereignisse.	14. Juli 2010
Bucket-Richtlinien	Bucket-Richtlinien sind ein Zugriffsmanagementsystem, mit dem Sie Zugriffsberechtigungen für Buckets, Objekte und Objektmengen festlegen. Diese Funktionalität ergänzt Zugriffsrichtlinien und ersetzt sie in vielen Fällen. Weitere Informationen finden Sie unter Bucket-Richtlinien und Benutzerrichtlinien.	6. Juli 2010
Path-Style-Syntax in allen Regionen verfügbar	Amazon S3 unterstützt jetzt die Pfad-Syntax für alle Buckets in der Region US Classic, oder wenn sich der Bucket innerhalb derselben Region befindet wie der Endpunkt der Anfrage. Weitere Informationen finden Sie unter Virtuelles Hosting.	9. Juni 2010
Neuer Endpunkt für Europa (Irland)	Amazon S3 bietet jetzt einen Endpunkt für Europa (Irland): `http://s3-eu-west-1.amazonaws.com` .	9. Juni 2010

Änderungen	Beschreibung	Datum
Konsole	Sie können Amazon S3 jetzt über die AWS Management Console verwenden. Sie können sich über alle Funktionen von Amazon S3 in der Konsole im Benutzerhandbuch des Amazon Simple Storage Service informieren.	9. Juni 2010
Reduzierte Redundanz	Amazon S3 ermöglicht Ihnen jetzt, Ihre Speicherkosten zu reduzieren, indem Objekte in Amazon S3 mit reduzierter Redundanz gespeichert werden. Weitere Informationen finden Sie unter Reduced Redundancy Storage.	12. Mai 2010
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region Asien-Pazifik (Singapur). Weitere Informationen finden Sie unter Buckets und Regionen.	28. April 2010
Objekt-Versioning	In dieser Version wird ein Objekt-Versioning eingeführt. Alle Objekte können jetzt einen Schlüssel und eine Version haben. Wenn Sie das Versioning für einen Bucket aktivieren, gibt Amazon S3 allen Objekten, die einem Bucket hinzugefügt werden, eine eindeutige Versions-ID. Diese Funktion ermöglicht Ihnen eine Wiederherstellung nach einem unbeabsichtigtem Überschreiben oder Löschen. Weitere Informationen finden Sie unter Versioning und Verwenden von Versioning.	8. Februar 2010
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region USA West (Nordkalifornien). Der neue Endpunkt für Anfragen an diese Region ist `s3-us-west-1.amazonaws.com` . Weitere Informationen finden Sie unter Buckets und Regionen.	2. Dezember 2009

Änderungen	Beschreibung	Datum
AWS SDK for .NET	AWS stellt jetzt Bibliotheken, Beispielcode, Tutorials und andere Ressourcen für Softwareentwickler bereit, die es vorziehen, Anwendungen mit .NET sprachspezifischen API-Operationen anstelle von REST oder SOAP zu erstellen. Diese Bibliotheken bieten grundlegende Funktionen (nicht in den REST oder SOAP APIs), wie beispielsweise die Anforderungsauthentifizierung, das erneute Absenden von Anforderungen und die Fehlerbehandlung, damit Ihnen der Einstieg erleichtert wird. Weitere Informationen zu sprachspezifischen Bibliotheken und Ressourcen finden Sie unter Entwickeln mit Amazon S3 unter Verwendung der AWS-SDKs und Explorer.	11. November 2009

AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar in der AWS-Glossar-Referenz.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.