aws

Benutzerhandbuch

EC2 Image Builder

EC2 Image Builder: Benutzerhandbuch

Table of Contents

Was ist EC2 Image Builder?

EC2 Image Builder ist ein vollständig verwalteter AWS-Service , mit dem Sie die Erstellung, Verwaltung und Bereitstellung benutzerdefinierter, sicherer und up-to-date Server-Images automatisieren können. Sie können die APIs AWS Management Console, oder verwendenAWS Command Line Interface, APIs um benutzerdefinierte Images in Ihrem zu erstellenAWS-Konto.

Sie besitzen die benutzerdefinierten Images, die Image Builder in Ihrem Konto erstellt. Sie können Pipelines konfigurieren, um Updates und System-Patches für die Images zu automatisieren, die Sie besitzen. Sie können auch einen eigenständigen Befehl ausführen, um ein Image mit den von Ihnen definierten Konfigurationsressourcen zu erstellen.

Der Image-Builder-Pipeline-Assistent kann Sie wie folgt durch die Schritte zum Erstellen eines benutzerdefinierten Images führen:

1. Wählen Sie ein Basis-Image für Ihre Anpassungen aus.
2. Fügen Sie Ihrem Basis-Image Software hinzu oder entfernen Sie sie.
3. Passen Sie Einstellungen und Skripts mit Build-Komponenten an.
4. Führen Sie ausgewählte Tests aus oder erstellen Sie benutzerdefinierte Testkomponenten.
5. Verteilen Sie AMIs an AWS-Regionen und AWS-Konten.
6. Wenn Ihre Image-Builder-Pipeline ein benutzerdefiniertes Amazon Machine Image (AMI) für die Verteilung erstellt, können Sie andere AWS-Konten, Organisationen und OUs autorisieren, es von Ihrem Konto aus zu starten. Ihrem Konto werden Gebühren in Rechnung gestellt, die mit dem AMI verbunden sind.

Image Builder lässt sich in Folgendes integrieren, AWS-Services um detaillierte Ereignismetriken, Protokollierung und Überwachung bereitzustellen. Diese Informationen helfen Ihnen dabei, Ihre Aktivitäten zu verfolgen, Image-Build-Probleme zu beheben und Automatisierungen basierend auf Ereignisbenachrichtigungen zu erstellen.

Abschnittsinhalte

- Preisgestaltung
- Verwandte AWS-Services

Funktionen von EC2 Image Builder

EC2 Image Builder bietet die folgenden Funktionen:

Erhöhen Sie die Produktivität und reduzieren Sie den Betrieb für die Erstellung von konformen - und -up-to-date Images

Image Builder reduziert den Arbeitsaufwand für die Erstellung und Verwaltung von Images in großem Umfang, indem Ihre Build-Pipelines automatisiert werden. Sie können Ihre Builds automatisieren, indem Sie Ihre Präferenz für den Build-Ausführungsplan angeben. Automation reduziert die Betriebskosten für die Wartung Ihrer Software mit den neuesten Betriebssystem-Patches.

Erhöhen der Serviceverfügbarkeit

Image Builder bietet Zugriff auf Testkomponenten, mit denen Sie Ihre Images vor der Bereitstellung testen können. Sie können auch benutzerdefinierte Testkomponenten mit AWS Task Orchestrator and Executor (AWSTOE) erstellen und diese verwenden. Image Builder verteilt Ihr Image nur, wenn alle konfigurierten Tests erfolgreich waren.

Erhöhen der Sicherheitsstandards für Bereitstellungen

Mit Image Builder können Sie Images erstellen, die unnötige Schwachstellen bei der Komponentensicherheit vermeiden. Sie können AWS Sicherheitseinstellungen anwenden, um sichere out-of-the-box Images zu erstellen, die branchenspezifische und interne Sicherheitskriterien erfüllen. Image Builder bietet auch Sammlungen von Einstellungen für Unternehmen in regulierten Branchen. Sie können diese Einstellungen verwenden, um schnell und einfach konforme Images für STIG-Standards zu erstellen. Eine vollständige Liste der über Image Builder verfügbaren STIG-Komponenten finden Sie unter Von Amazon verwaltete STIG-Harding-Komponenten für EC2 Image Builder.

Zentralisierte Durchsetzung und Nachverfolgung der Herkunft

Mithilfe der integrierten Integrationen mit können Sie mit Image Builder Richtlinien erzwingenAWS Organizations, die Konten auf die Ausführung von Instances nur von genehmigten AMIs beschränken.

Vereinfachte gemeinsame Nutzung von Ressourcen über hinweg AWS-Konten

EC2 Image Builder lässt sich in AWS Resource Access Manager (AWS RAM) integrieren, damit Sie bestimmte Ressourcen mit einem AWS-Konto oder über teilen könnenAWS Organizations. EC2 Image Builder-Ressourcen, die freigegeben werden können, sind:

- Komponenten
- Bilder
- Image-Rezepte
- Container-Rezepte

Weitere Informationen finden Sie unter EC2 Image Builder-Ressourcen freigeben.

Unterstützte Betriebssysteme

Image Builder unterstützt die folgenden Betriebssystemversionen:

Betriebssystem/Verteilung	Unterstützte Versionen
Amazon Linux	2 und 2023
CentOS	7 und 8
CentOS Stream	8
Red Hat Enterprise Linux (RHEL)	7 und 8
SUSE Linux Enterprise Server (SUSE)	12 und 15
Ubuntu	18.04 LTS, 20.04 LTS und 22.04 LTS
Windows Server	2012 R2, 2016, 2019 und 2022

Unterstützte Image-Formate

Für Ihre benutzerdefinierten AMI-Images können Sie ein vorhandenes AMI als Ausgangspunkt auswählen. Für Docker-Container-Images können Sie aus öffentlichen Images wählen, die auf

gehostet werden DockerHub, aus vorhandenen Container-Images in Amazon ECR oder aus von Amazon verwalteten Container-Images.

Konzepte

Die folgenden Begriffe und Konzepte sind für Ihr Verständnis und Ihre Verwendung von EC2 Image Builder von zentraler Bedeutung.

AMI

Ein Amazon Machine Image (AMI) ist die grundlegende Bereitstellungseinheit in Amazon EC2 und ist einer der Arten von Images, die Sie mit Image Builder erstellen können. Ein AMI ist ein vorkonfiguriertes Abbild für virtuelle Maschinen, das das Betriebssystem (OS) und die vorinstallierte Software zur Bereitstellung von EC2-Instances enthält. Weitere Informationen finden Sie unter Amazon Machine Images (AMI).

Image-Pipeline

Eine Image-Pipeline bietet ein Automatisierungs-Framework zum Erstellen sicherer AMIs und Container-Images auf AWS. Die Image-Builder-Image-Pipeline ist einem Image-Rezept oder Container-Rezept zugeordnet, das die Build-, Validierungs- und Testphasen für einen Image-Build-Lebenszyklus definiert.

Eine Image-Pipeline kann einer Infrastrukturkonfiguration zugeordnet werden, die festlegt, wo Ihr Image erstellt wird. Sie können Attribute wie Instance-Typ, Subnetze, Sicherheitsgruppen, Protokollierung und andere infrastrukturbezogene Konfigurationen definieren. Sie können Ihre Image-Pipeline auch einer Verteilungskonfiguration zuordnen, um festzulegen, wie Sie Ihr Image bereitstellen möchten.

Verwaltetes Image

Ein verwaltetes Image ist eine Ressource in Image Builder, die aus einem AMI oder Container-Image sowie Metadaten wie Version und Plattform besteht. Das verwaltete Image wird von Image-Builder-Pipelines verwendet, um zu bestimmen, welches Basis-Image für den Build verwendet werden soll. In diesem Handbuch werden verwaltete Images manchmal als „Images“ bezeichnet. Ein Image ist jedoch nicht dasselbe wie ein AMI.

Image-Rezept

Ein Image-Builder-Image-Rezept ist ein Dokument, das das Basis-Image und die Komponenten definiert, die auf das Basis-Image angewendet werden, um die gewünschte Konfiguration für

das Ausgabe-AMI-Image zu erzeugen. Sie können ein Image-Rezept verwenden, um Builds zu duplizieren. Image Builder-Image-Rezepte können mithilfe des Konsolen-Assistenten, der AWS CLI oder der API freigegeben, verzweigt und bearbeitet werden. Sie können Image-Rezepte mit Ihrer Versionskontrollsoftware verwenden, um gemeinsam nutzbare, versionierte Image-Rezepte zu verwalten.

Container-Rezept

Ein Image-Builder-Container-Rezept ist ein Dokument, das das Basis-Image und die Komponenten definiert, die auf das Basis-Image angewendet werden, um die gewünschte Konfiguration für das Ausgabe-Container-Image zu erzeugen. Sie können ein Container-Rezept verwenden, um Builds zu duplizieren. Sie können Image-Builder-Image-Rezepte mithilfe des Konsolenassistenten, der oder der API freigebenAWS CLI, verzweigen und bearbeiten. Sie können Container-Rezepte mit Ihrer Versionskontrollsoftware verwenden, um gemeinsam nutzbare, versionierte Container-Rezepte zu verwalten.

Basis-Image

Das Basis-Image ist das ausgewählte Image und das Betriebssystem, die in Ihrem Image- oder Container-Rezeptdokument verwendet werden, zusammen mit den Komponenten. Das Basis-Image und die Komponentendefinitionen erzeugen zusammen die gewünschte Konfiguration für das Ausgabe-Image.

Komponenten

Eine Komponente definiert die Reihenfolge der Schritte, die erforderlich sind, um entweder eine Instance vor der Image-Erstellung anzupassen (eine Build-Komponente) oder um eine Instance zu testen, die über das erstellte Image gestartet wurde (eine Testkomponente).

Eine Komponente wird aus einem deklarativen Klartext-YAML- oder JSON-Dokument erstellt, das die Laufzeitkonfiguration zum Erstellen und Validieren oder Testen einer Instance beschreibt, die von Ihrer Pipeline erstellt wird. Komponenten werden auf der Instance mit einer Komponentenverwaltungsanwendung ausgeführt. Die Komponentenverwaltungsanwendung analysiert die Dokumente und führt die gewünschten Schritte aus.

Nachdem sie erstellt wurden, werden eine oder mehrere Komponenten mithilfe eines Image-Rezepts oder Container-Rezepts gruppiert, um den Plan zum Erstellen und Testen eines virtuellen Maschinen- oder Container-Images zu definieren. Sie können öffentliche Komponenten verwenden, die sich im Besitz von befinden und von verwaltet werdenAWS, oder Sie können Ihre eigenen erstellen.

Weitere Informationen zu -Komponenten finden Sie unter AWS Task Orchestrator and Executor Komponentenmanager.

Komponentendokument

Ein deklaratives Klartext-YAML- oder JSON-Dokument, das die Konfiguration für eine Anpassung beschreibt, die Sie auf Ihr Image anwenden können. Das -Dokument wird verwendet, um eine Build- oder Testkomponente zu erstellen.

Laufzeitphasen

EC2 Image Builder hat zwei Laufzeitphasen: erstellen und testen. Jede Laufzeitphase hat eine oder mehrere Phasen mit Konfiguration, die durch das Komponentendokument definiert wird.

Konfigurationsphasen

Die folgende Liste zeigt die Phasen, die während der Build- und Testphasen ausgeführt werden:

Build-Phase:

Build-Phase

Eine Image-Pipeline beginnt mit der Build-Phase der Build-Phase, in der sie ausgeführt wird. Das Basis-Image wird heruntergeladen, und die Konfiguration, die für die Build-Phase der Komponente angegeben ist, wird angewendet, um eine Instance zu erstellen und zu starten.

Validieren der Phase

Nachdem Image Builder die Instance gestartet und alle Anpassungen der Build-Phase angewendet hat, beginnt die Validierungsphase. Während dieser Phase stellt Image Builder sicher, dass alle Anpassungen basierend auf der Konfiguration, die die Komponente für die Validierungsphase angibt, wie erwartet funktionieren. Wenn die Instance-Validierung erfolgreich ist, stoppt Image Builder die Instance, erstellt ein Image und fährt dann mit der Testphase fort.

Testphase:

Testphase

Während dieser Phase startet Image Builder eine Instance aus dem Image, das es erstellt hat, nachdem die Validierungsphase erfolgreich abgeschlossen wurde. Image Builder führt in dieser Phase Testkomponenten aus, um zu überprüfen, ob die Instance fehlerfrei ist und wie erwartet funktioniert.

Container-Host-Testphase

Nachdem Image Builder die Testphase für alle Komponenten ausgeführt hat, die Sie im Container-Rezept ausgewählt haben, führt Image Builder diese Phase für Container-Workflows aus. Die Container-Host-Testphase kann zusätzliche Tests ausführen, die die Containerverwaltung und benutzerdefinierte Laufzeitkonfigurationen validieren.

Workflow

Workflows definieren die Reihenfolge der Schritte, die Image Builder ausführt, wenn es ein neues Image erstellt. Alle Images verfügen über Build- und Test-Workflows. Container verfügen über einen zusätzlichen Workflow für die Verteilung.

Workflow-Typen

`BUILD`

Behandelt die Konfiguration der Build-Stufe für jedes erstellte Image.

`TEST`

Behandelt die Teststufenkonfiguration für jedes erstellte Image.

`DISTRIBUTION`

Behandelt den Verteilungs-Workflow für Container-Images.

Preisgestaltung

Die Verwendung von EC2 Image Builder zum Erstellen von benutzerdefinierten AMI- oder Container-Images ist kostenlos. Die Standardpreise gelten jedoch für andere -Services, die im Prozess verwendet werden. Die folgende Liste enthält die Nutzung einiger AWS-Services, die je nach Konfiguration Kosten verursachen können, wenn Sie Ihr benutzerdefiniertes AMI oder Ihre Container-Images erstellen, erstellen, speichern und verteilen.

- Starten einer EC2-Instance
- Speichern von Protokollen in Amazon S3
- Validieren von Bildern mit Amazon Inspector
- Speichern von Amazon-EBS-Snapshots für Ihre AMIs
- Speichern von Container-Images in Amazon ECR

- Verschieben und Abrufen von Container-Images in und aus Amazon ECR
- Wenn Systems Manager Advanced Tier aktiviert ist und Amazon EC2-Instances mit On-Premises-Aktivierung ausgeführt werden, werden Ihnen möglicherweise Ressourcen über Systems Manager in Rechnung gestellt

Verwandte AWS-Services

EC2 Image Builder verwendet andere AWS-Services, um Images zu erstellen. Abhängig von Ihrem Image-Builder-Image-Rezept oder Ihrer Container-Rezeptkonfiguration können die folgenden Services verwendet werden.

AWS License Manager

AWS License Manager ermöglicht Ihnen das Erstellen und Anwenden von Lizenzkonfigurationen aus einem Konto-Lizenzkonfigurationsspeicher. Für jedes AMI können Sie Image Builder verwenden, um einer bereits vorhandenen Lizenzkonfiguration anzufügen, auf die Ihr im Rahmen des Image-Builder-Workflows Zugriff AWS-Konto hat. Lizenzkonfigurationen können nur auf AMIs angewendet werden. Image Builder kann nur bereits vorhandene Lizenzkonfigurationen verwenden und Lizenzkonfigurationen nicht direkt erstellen oder ändern. License Manager-Einstellungen werden nicht über hinweg repliziert, AWS-Regionen die in Ihrem Konto aktiviert werden müssen, z. B. zwischen den Regionen `ap-east-1` (Asien-Pazifik: Hong Kong) und `me-south-1` (Middle East: Bahrain).

AWS Organizations

AWS Organizations Mit können Sie Service-Kontrollrichtlinien (SCP) auf Konten in Ihrer Organisation anwenden. Sie können einzelne Richtlinien erstellen, verwalten, aktivieren und deaktivieren. Ähnlich wie bei allen anderen AWS Artefakten und Services berücksichtigt Image Builder die in definierten RichtlinienAWS Organizations. AWS bietet Vorlagen-SCPs für gängige Szenarien, z. B. die Durchsetzung von Einschränkungen für Mitgliedskonten, um Instances nur mit genehmigten AMIs zu starten.

Amazon Inspector

Image Builder verwendet Amazon Inspector als Standard-Agent für Schwachstellenscans, um Sicherheitsgrundlagen für Amazon Linux 2, Windows Server 2012 und Windows Server 2016 einzurichten. Weitere Informationen finden Sie unter Was ist Amazon Inspector?

AWS Resource Access Manager

AWS Resource Access Manager Mit (AWS RAM) können Sie Ihre -Ressourcen mit einem beliebigen AWS-Konto oder über teilenAWS Organizations. Wenn Sie mehrere habenAWS-Konten, können Sie Ressourcen zentral erstellen und verwenden, AWS RAM um diese Ressourcen für andere Konten freizugeben. EC2 Image Builder ermöglicht die Freigabe für die folgenden Ressourcen: Komponenten, Images und Image-Rezepte. Weitere Informationen zu AWS RAM finden Sie im AWS Resource Access Manager-Benutzerhandbuch. Informationen zum Freigeben von Image-Builder-Ressourcen finden Sie unter EC2 Image Builder-Ressourcen freigeben.

Amazon CloudWatch -Protokolle

Sie können Amazon CloudWatch Logs verwenden, um Ihre Protokolldateien von EC2-Instances, AWS CloudTrail, Amazon Route 53 und anderen Quellen zu überwachen, zu speichern und darauf zuzugreifen.

Amazon Elastic Container Registry (Amazon ECR)

Amazon ECR ist ein verwalteter AWS Container-Image-Registry-Service, der sicher, skalierbar und zuverlässig ist. Container-Images, die Sie mit Image Builder erstellen, werden in Amazon ECR in Ihrer Quellregion (in der Ihr Build ausgeführt wird) und in allen Regionen gespeichert, in denen Sie das Container-Image verteilen. Weitere Informationen zu Amazon ECR finden Sie im Amazon Elastic Container Registry-Benutzerhandbuch.

Funktionsweise von EC2 Image Builder

Wenn Sie den Pipeline-Konsolenassistenten von EC2 Image Builder verwenden, um ein benutzerdefiniertes Image zu erstellen, führt Sie ein Assistent durch die folgenden Schritte.

1. Pipeline-Details angeben – Geben Sie Informationen zu Ihrer Pipeline ein, z. B. einen Namen, eine Beschreibung, Tags und einen Zeitplan für die Ausführung automatisierter Builds. Sie können bei Bedarf manuelle Builds auswählen.
2. Rezept auswählen – Wählen Sie zwischen dem Erstellen eines AMI oder dem Erstellen eines Container-Images. Für beide Arten von Ausgabe-Images geben Sie einen Namen und eine Version für Ihr Rezept ein, wählen ein Basis-Image aus und wählen Komponenten aus, die zum Erstellen und Testen hinzugefügt werden sollen. Sie können auch die automatische Versionsverwaltung wählen, um sicherzustellen, dass Sie immer die neueste verfügbare Betriebssystemversion (OS) für Ihr Basis-Image verwenden. Container-Rezepte definieren zusätzlich Dockerfiles und das Amazon ECR-Ziel-Repository für Ihr Docker-Container-Image.

   > **Note**
   >
   > Komponenten sind die Bausteine, die von einem Image-Rezept oder einem Container-Rezept verbraucht werden. Zum Beispiel Pakete für die Installation, Schritte zur Erhöhung der Sicherheit und Tests. Das ausgewählte Basis-Image und die Komponenten bilden ein Image-Rezept.

3. Infrastrukturkonfiguration definieren – Image Builder startet EC2-Instances in Ihrem Konto, um Images anzupassen und Validierungstests durchzuführen. Die Konfigurationseinstellungen für die Infrastruktur geben Infrastrukturdetails für die Instances an, die AWS-Konto während des Build-Prozesses in Ihrem ausgeführt werden.
4. Verteilungseinstellungen definieren – Wählen Sie die AWS Regionen aus, an die Ihr Image verteilt werden soll, nachdem der Build abgeschlossen ist und alle seine Tests bestanden hat. Die Pipeline verteilt Ihr Image automatisch an die Region, in der der Build ausgeführt wird, und Sie können die Image-Verteilung für andere Regionen hinzufügen.

Die Images, die Sie aus Ihrem benutzerdefinierten Basis-Image erstellen, befinden sich in Ihrem AWS-Konto. Sie können Ihre Image-Pipeline so konfigurieren, dass aktualisierte und gepatchte Versionen Ihres Images erstellt werden, indem Sie einen Build-Zeitplan eingeben. Wenn der Build abgeschlossen ist, können Sie eine Benachrichtigung über Amazon Simple Notification Service

(SNS) erhalten. Neben der Erstellung eines endgültigen Images generiert der Image-Builder-Konsolenassistent ein Rezept, das mit vorhandenen Versionskontrollsystemen und Pipelines für kontinuierliche Integration/kontinuierliche Bereitstellung (CI/CD) für wiederholbare Automatisierung verwendet werden kann. Sie können neue Versionen Ihres Rezepts freigeben und erstellen.

Abschnittsinhalte

AMI-Elemente

Ein Amazon Machine Image (AMI) ist ein vorkonfiguriertes VM-Image (Virtual Machine), das das Betriebssystem und die Software für die Bereitstellung von EC2-Instances enthält.

Ein AMI enthält die folgenden Elemente:

- Eine Vorlage für das Stamm-Volume der VM. Wenn Sie eine Amazon EC2-VM starten, enthält das Root-Gerät-Volume das Image zum Starten der Instance. Wenn Instance-Speicher verwendet wird, ist das Root-Gerät ein Instance-Speicher-Volume, das aus einer Vorlage in Amazon S3 erstellt wurde. Weitere Informationen finden Sie unter Amazon EC2 Root Device Volume.
- Wenn Amazon EBS verwendet wird, ist das Root-Gerät ein EBS-Volume, das aus einem EBS-Snapshot erstellt wurde.
- Startberechtigungen, die die bestimmenAWS-Konten, die VMs mit dem AMI starten können.
- Blockgerät-Zuweisungsdaten, die die Volumes angeben, die nach dem Start an die Instance angefügt werden sollen.
- Eine eindeutige Ressourcen-ID für jede Region und für jedes Konto.

- Metadatennutzlasten wie Tags und Eigenschaften wie Region, Betriebssystem, Architektur, Root-Gerätetyp, Anbieter, Startberechtigungen, Speicher für das Root-Gerät und Signaturstatus.
- Eine AMI-Signatur für Windows-Images zum Schutz vor unbefugter Manipulation. Weitere Informationen finden Sie unter Instance-Identitätsdokumente.

Standardkontingente

Informationen zum Anzeigen der Standardkontingente für Image Builder finden Sie unter Image-Builder-Endpunkte und -Kontingente.

Regionen und Endpunkte von AWS

Informationen zum Anzeigen der Service-Endpunkte für Image Builder finden Sie unter Image-Builder-Endpunkte und -Kontingente.

Komponentenverwaltung

EC2 Image Builder verwendet eine Komponentenverwaltungsanwendung AWS Task Orchestrator and Executor (AWSTOE), mit der Sie komplexe Workflows orchestrieren, Systemkonfigurationen ändern und Ihre Systeme mit YAML-basierten Skriptkomponenten testen können. Da es sich um eine eigenständige Anwendung AWSTOE handelt, ist keine zusätzliche Einrichtung erforderlich. Es kann in jeder Cloud-Infrastruktur und On-Premises ausgeführt werden. Informationen zu den ersten Schritten mit AWSTOE als eigenständige Anwendung finden Sie unter Erste Schritte mit AWSTOE.

Image Builder verwendet AWSTOE, um alle Aktivitäten auf der Instance durchzuführen. Dazu gehören das Erstellen und Validieren Ihres Images, bevor Sie einen Snapshot erstellen, und das Testen des Snapshots, um sicherzustellen, dass es wie erwartet funktioniert, bevor Sie das endgültige Image erstellen. Weitere Informationen darüber, wie Image Builder AWSTOE zur Verwaltung seiner Komponenten verwendet, finden Sie unter Verwalten von Komponenten mit Image Builder. Weitere Informationen zum Erstellen von Komponenten mit AWSTOEfinden Sie unter AWS Task Orchestrator and Executor Komponentenmanager.

Image-Tests

Sie können AWSTOE Testkomponenten verwenden, um Ihr Image zu validieren und sicherzustellen, dass es wie erwartet funktioniert, bevor Sie das endgültige Image erstellen.

Im Allgemeinen besteht jede Testkomponente aus einem YAML-Dokument, das ein Testskript, eine Testbinärdatei und Testmetadaten enthält. Das Testskript enthält die Orchestrierungsbefehle zum Starten der Testbinärdatei, die in jeder vom Betriebssystem unterstützten Sprache geschrieben werden kann. Beendigungsstatuscodes geben das Testergebnis an. Testmetadaten beschreiben den Test und sein Verhalten, z. B. den Namen, die Beschreibung, die Pfade zum Testen der Binärdatei und die erwartete Dauer.

Semantische Versionsverwaltung

Image Builder verwendet semantische Versionsverwaltung, um Ressourcen zu organisieren und sicherzustellen, dass sie eindeutige IDs haben. Die semantische Version hat vier Knoten:

<major>.*<minor>*.*<patch>*/<build>

Sie können Werte für die ersten drei zuweisen und nach allen filtern.

Die semantische Versionsverwaltung ist wie folgt im Amazon-Ressourcennamen (ARN) jedes Objekts auf der Ebene enthalten, die für dieses Objekt gilt:

1. Versionslose ARNs und Namens-ARNs enthalten keine spezifischen Werte in einem der Knoten. Die Knoten werden entweder vollständig ausgeschaltet oder als Platzhalter angegeben, z. B. x.x.x.
2. Versions-ARNs haben nur die ersten drei Knoten: <major>.<minor>.<patch>
3. Build-Versions-ARNs haben alle vier Knoten und verweisen auf einen bestimmten Build für eine bestimmte Version eines Objekts.

Zuweisung: Für die ersten drei Knoten können Sie einen beliebigen positiven Ganzzahlwert, einschließlich Null, mit einer Obergrenze von 2^30-1 oder 1073741823 für jeden Knoten zuweisen. Image Builder weist die Build-Nummer automatisch dem vierten Knoten zu.

Muster: Sie können jedes numerische Muster verwenden, das den Zuweisungsanforderungen für die Knoten entspricht, die Sie zuweisen können. Sie können beispielsweise ein Muster für die Softwareversion wie 1.0.0 oder ein Datum wie 2021.01.01 wählen.

Auswahl: Mit der semantischen Versionsverwaltung haben Sie die Flexibilität, Platzhalter (x) zu verwenden, um die neuesten Versionen oder Knoten anzugeben, wenn Sie das Basis-Image oder die Komponenten für Ihr Rezept auswählen. Wenn Sie in einem Knoten einen Platzhalter verwenden, müssen alle Knoten rechts vom ersten Platzhalter ebenfalls Platzhalter sein.

Bei den folgenden aktuellen Versionen: 2.2.4, 1.7.8 und 1.6.8 führt die Versionsauswahl mit Platzhaltern zu den folgenden Ergebnissen:

- `x.x.x` = 2.2.4
- `1.x.x` = 1.7.8
- `1.6.x` = 1.6.8
- `x.2.x` ist ungültig und erzeugt einen Fehler
- `1.x.8` ist ungültig und erzeugt einen Fehler

Erstellte Ressourcen

Wenn Sie eine Pipeline erstellen, werden keine Ressourcen außerhalb von Image Builder erstellt, es sei denn, Folgendes gilt:

- Wenn ein Image über den Pipeline-Zeitplan erstellt wird
- Wenn Sie im Menü Aktionen in der Image-Builder-Konsole Pipeline ausführen auswählen
- Wenn Sie einen dieser Befehle über die API oder ausführenAWS CLI: StartImagePipelineExecution oder CreateImage

Die folgenden Ressourcen werden während des Image-Build-Prozesses erstellt:

AMI-Image-Pipelines

- EC2-Instance (vorübergehend)
- Systems Manager Inventory Association (über Systems Manager State Manager, wenn aktiviert `EnhancedImageMetadata` ist) auf der EC2-Instance
- Amazon EC2-AMI
- Der Amazon-EBS-Snapshot, der dem Amazon EC2-AMI zugeordnet ist

Container-Image-Pipelines

- Docker-Container, der auf einer EC2-Instance ausgeführt wird (temporär)
- Systems Manager Inventory Association (über Systems Manager State Manager) `EnhancedImageMetadata` ist aktiviert) auf der EC2-Instance
- Docker-Container-Image

- Dockerfile

Nachdem das Image erstellt wurde, werden alle temporären Ressourcen gelöscht.

Distribution

EC2 Image Builder kann AMIs oder Container-Images an jede AWS Region verteilen. Das Image wird in jede Region kopiert, die Sie in dem Konto angeben, das zum Erstellen des Images verwendet wurde.

Für AMI-Ausgabeabbilder können Sie AMI-Startberechtigungen definieren, um zu steuern, welche EC2-Instances mit dem erstellten AMI starten AWS-Konten dürfen. Sie können das Image beispielsweise privat, öffentlich machen oder für bestimmte Konten freigeben. Wenn Sie sowohl das AMI an andere Regionen verteilen als auch Startberechtigungen für andere Konten definieren, werden die Startberechtigungen an die AMIs in allen Regionen weitergegeben, in denen das AMI verteilt ist.

Sie können Ihr AWS Organizations Konto auch verwenden, um Einschränkungen für Mitgliedskonten durchzusetzen, um Instances nur mit genehmigten und konformen AMIs zu starten. Weitere Informationen finden Sie unter Verwalten der AWS-Konten in Ihrer Organisation.

Um Ihre Verteilungseinstellungen mithilfe der Image-Builder-Konsole zu aktualisieren, führen Sie die Schritte zu Erstellen einer neuen Image-Rezeptversion (Konsole)oder ausErstellen einer neuen Container-Rezeptversion mit der Konsole.

Freigeben von Ressourcen

Informationen zum Freigeben von Komponenten, Rezepten oder Bildern für andere Konten oder innerhalb AWS Organizationsvon finden Sie unter EC2 Image Builder-Ressourcen freigeben.

-Compliance

Für CIS verwendet EC2 Image Builder Amazon Inspector, um Bewertungen auf Risiken, Schwachstellen und Abweichungen von bewährten Methoden und Compliance-Standards durchzuführen. Image Builder bewertet beispielsweise unbeabsichtigte Netzwerkzugänglichkeit, nicht gepatchte CVEs, öffentliche Internetverbindung und Aktivierung der Remote-Root-Anmeldung. Amazon Inspector wird als Testkomponente angeboten, die Sie Ihrem Image-Rezept

hinzufügen können. Weitere Informationen zu Amazon Inspector finden Sie im Amazon Inspector-Benutzerhandbuch. Zum Verstärken validiert EC2 Image Builder mit STIG. Eine vollständige Liste der über Image Builder verfügbaren STIG-Komponenten finden Sie unter Von Amazon verwaltete STIG-Harding-Komponenten für EC2 Image Builder. Weitere Informationen finden Sie unter Center for Internet Security (CIS) Benchmarks.

Erste Schritte mit EC2 Image Builder

Dieses Kapitel hilft Ihnen, Ihre Umgebung einzurichten und zum ersten Mal eine automatisierte Image-Pipeline oder Container-Pipeline zu erstellen, indem Sie den Assistenten zum Erstellen einer Image-Pipeline von EC2 Image Builder verwenden.

Inhalt

Voraussetzungen

Überprüfen Sie die folgenden Voraussetzungen, um eine Image-Pipeline mit EC2 Image Builder zu erstellen. Sofern nicht anders angegeben, sind die Voraussetzungen für alle Arten von Pipelines erforderlich.

Serviceverknüpfte Rolle EC2 Image Builder

EC2 Image Builder verwendet eine serviceverknüpfte Rolle, um anderen -AWSServices in Ihrem Namen Berechtigungen zu erteilen. Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie Ihre erste Image-Builder-Ressource in der -AWSManagementkonsoleAWS CLI, der oder der -AWSAPI erstellen, erstellt Image Builder die serviceverknüpfte Rolle für Sie. Weitere Informationen zur serviceverknüpften Rolle, die Image Builder in Ihrem Konto erstellt, finden Sie unter Verwenden von serviceverknüpften Rollen für EC2 Image Builder.

Konfigurationsanforderungen

- Image Builder unterstützt AWS PrivateLink. Weitere Informationen zum Konfigurieren von VPC-Endpunkten für Image Builder finden Sie unter EC2 Image Builder und Schnittstellen-VPC-Endpunkte (AWS PrivateLink).
- Image Builder unterstützt EC2-Classic .
- Die Instances, die Image Builder zum Erstellen von Container-Images verwendet, müssen Internetzugang haben, um AWS CLI von Amazon S3 herunterzuladen und gegebenenfalls ein

Basis-Image aus dem Docker Hub-Repository herunterzuladen. Image Builder verwendet die AWS CLI, um die Dockerfile aus dem Container-Rezept abzurufen, wo sie als Daten gespeichert wird.

- Die Instances, die Image Builder zum Erstellen von Images und zum Ausführen von Tests verwendet, müssen Zugriff auf den Systems Manager-Service haben. Die Installationsanforderungen hängen von Ihrem Betriebssystem ab.

  Um die Installationsanforderungen für Ihr Basis-Image anzuzeigen, wählen Sie die Registerkarte aus, die Ihrem Basis-Image-Betriebssystem entspricht.

  Linux

  Für Amazon EC2 Linux-Instances installiert Image Builder den Systems Manager Agent auf der Build-Instance, falls er noch nicht vorhanden ist, und entfernt ihn, bevor das Image erstellt wird.

  Windows

  Image Builder installiert den Systems Manager Agent nicht auf Amazon EC2 Windows Server-Build-Instances. Wenn Ihr Basis-Image nicht mit dem Systems Manager Agent vorinstalliert wurde, müssen Sie eine Instance aus Ihrem Quell-Image starten, Systems Manager manuell auf der Instance installieren und ein neues Basis-Image aus Ihrer Instance erstellen.

  Informationen zur manuellen Installation des Systems-Manager-Agenten auf Ihrer Amazon EC2-Windows-Server-Instance finden Sie unter Manuelle Installation des Systems-Manager-Agenten auf EC2-Instances für Windows Server im AWS Systems Manager - Benutzerhandbuch.

Container-Repository (Container-Image-Pipelines)

Für Container-Image-Pipelines definiert das Rezept die Konfiguration für die Docker-Images, die im Ziel-Container-Repository erstellt und gespeichert werden. Sie müssen das Ziel-Repository erstellen, bevor Sie das Container-Rezept für Ihr Docker-Image erstellen.

Image Builder verwendet Amazon ECR als Ziel-Repository für Container-Images. Um ein Amazon ECR-Repository zu erstellen, führen Sie die unter Erstellen eines Repositorys im Benutzerhandbuch für Amazon Elastic Container Registry beschriebenen Schritte aus.

AWS Identity and Access Management (IAM)

Die IAM-Rolle, die Sie Ihrem Instance-Profil zuordnen, muss über Berechtigungen zum Ausführen der Build- und Testkomponenten verfügen, die in Ihrem Image enthalten sind. Die folgenden IAM-Rollenrichtlinien müssen an die IAM-Rolle angehängt werden, die dem Instance-Profil zugeordnet ist:

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- AmazonSSMManagedInstanceCore

Wenn Sie die Protokollierung konfigurieren, muss das in Ihrer Infrastrukturkonfiguration angegebene Instance-Profil über `s3:PutObject` Berechtigungen für den Ziel-Bucket () verfügen`arn:aws:s3:::BucketName/*`. Beispielsweise:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::bucket-name/*"
        }
    ]
}
```

Richtlinie anfügen

Die folgenden Schritte führen Sie durch den Prozess zum Anfügen der IAM-Richtlinien an eine IAM-Rolle, um die oben genannten Berechtigungen zu erteilen.

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Richtlinien aus.
3. Filtern Sie die Liste der Richtlinien mit EC2InstanceProfileForImageBuilder
4. Wählen Sie den Aufzählungspunkt neben der Richtlinie aus und wählen Sie in der Dropdownliste Richtlinienaktionen die Option Anfügen aus.

5. Wählen Sie den Namen der IAM-Rolle aus, an die die Richtlinie angehängt werden soll.
6. Wählen Sie Richtlinie anfügen aus.
7. Wiederholen Sie die Schritte 3-6 für die AmazonSSMManagedInstanceCore-Richtlinien EC2InstanceProfileForImageBuilderECRContainerBuilds und .

> **Note**
>
> Wenn Sie ein mit Image Builder erstelltes Image in ein anderes Konto kopieren möchten, müssen Sie die `EC2ImageBuilderDistributionCrossAccountRole` Rolle in allen Zielkonten erstellen und die Ec2ImageBuilderCrossAccountDistributionAccess-Richtlinie verwaltete Richtlinie an die Rolle anfügen. Weitere Informationen finden Sie unter EC2 Image Builder-Ressourcen freigeben.

Zugriff auf EC2 Image Builder

Sie können EC2 Image Builder über eine der folgenden Schnittstellen verwalten.

- Landingpage der EC2 Image Builder-Konsole . Von der EC2 Image Builder-Konsole aus.
- AWS Command Line Interface (AWS CLI) Sie können die verwendenAWS CLI, um auf AWS -API-Operationen zuzugreifen. Weitere Informationen finden Sie unter Installieren der -AWSBefehlszeilenschnittstelle im AWS Command Line Interface-Benutzerhandbuch.
- AWS Tools für SDKs . Sie können AWS SDKs und Tools verwenden, um mit Ihrer bevorzugten Sprache auf Image Builder zuzugreifen und ihn zu verwalten.

Erstellen einer Image-Pipeline mit dem EC2 Image Builder-Konsolenassistenten

Dieses Tutorial führt Sie durch das Erstellen einer automatisierten Pipeline zum Erstellen und Verwalten eines benutzerdefinierten EC2 Image Builder-Images mithilfe des Assistenten zum Erstellen einer Image-Pipeline. Damit Sie die Schritte effizient ausführen können, werden Standardeinstellungen verwendet, wenn sie verfügbar sind, und optionale Abschnitte werden übersprungen.

Erstellen eines Image-Pipeline-Workflows

Schritt 1: Festlegen von Pipeline-Details

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um mit der Erstellung Ihrer Pipeline zu beginnen, wählen Sie Image-Pipeline erstellen aus.
3. Geben Sie im Abschnitt Allgemein Ihren Pipeline-Namen ein (erforderlich).

   > **Tip**
   >
   > Die erweiterte Metadatensammlung ist standardmäßig aktiviert. Um die Kompatibilität zwischen Komponenten und Basis-Images sicherzustellen, lassen Sie sie aktiviert.

4. Im Abschnitt Zeitplan erstellen können Sie die Standardwerte für die Zeitplanoptionen beibehalten. Beachten Sie, dass die für den Standardzeitplan angezeigte Zeitzone die Universal Coordinated Time (UTC) ist. Weitere Informationen zur UTC-Zeit und zum Ermitteln des Offsets für Ihre Zeitzone finden Sie unter Zeitzonenabkürzungen – Weltweite Liste.

   Wählen Sie für Einstellungen für Abhängigkeitsaktualisierungen die Option Pipeline zum geplanten Zeitpunkt ausführen aus, wenn es Abhängigkeitsaktualisierungen gibt. Diese Einstellung führt dazu, dass Ihre Pipeline vor dem Start des Builds nach Updates sucht. Wenn es keine Updates gibt, wird der geplante Pipeline-Build übersprungen.

   > **Note**
   >
   > Um sicherzustellen, dass Ihre Pipeline Abhängigkeitsaktualisierungen erkennt und wie erwartet erstellt, müssen Sie semantische Versionsverwaltung (x.x.x) für Ihr Basis-Image und Ihre Komponenten verwenden. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

5. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 2: Rezept auswählen

1. Image Builder verwendet standardmäßig das vorhandene Rezept im Abschnitt Rezept. Wählen Sie zum ersten Mal die Option Neues Rezept erstellen aus.
2. Wählen Sie im Abschnitt Image-Typ die Option Amazon Machine Image (AMI) ,um eine Image-Pipeline zu erstellen, die ein AMI erzeugt und verteilt.
3. Geben Sie im Abschnitt Allgemein die folgenden Pflichtfelder ein:

   - Name – Ihr Rezeptname
   - Version – Ihre Rezeptversion (verwenden Sie das Format <major>.<minor>.<patch>, wobei major, minor und patch Ganzzahlwerte sind). Neue Rezepte beginnen im Allgemeinen mit `1.0.0`.
4. Behalten Sie im Abschnitt Quellbild die Standardwerte für Image auswählen, Image-Betriebssystem (OS) und Image-Ursprung bei. Dies führt zu einer Liste der von Amazon verwalteten Amazon Linux 2-AMIs, aus denen Sie für Ihr Basis-Image wählen können.

   a. Wählen Sie in der Dropdownliste Image name ein Image aus.

   b. Behalten Sie die Standardeinstellung für Optionen für die automatische Versionsverwaltung bei (Verwenden Sie die neueste verfügbare Betriebssystemversion).

   > **Note**
   >
   > Diese Einstellung stellt sicher, dass Ihre Pipeline semantische Versionsverwaltung für das Basis-Image verwendet, um Abhängigkeitsaktualisierungen für automatisch geplante Aufträge zu erkennen. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.
5. Behalten Sie im Abschnitt Instance-Konfiguration die Standardwerte für den Systems Manager-Agent bei. Dies führt dazu, dass Image Builder den Systems-Manager-Agenten nach Abschluss der Builds und Tests beibehält, um den Systems-Manager-Agenten in Ihr neues Image aufzunehmen.

   Lassen Sie die Benutzerdaten für dieses Tutorial leer. Sie können diesen Bereich zu anderen Zeiten verwenden, um Befehle bereitzustellen, oder ein Befehlsskript, das beim Starten Ihrer Build-Instance ausgeführt werden soll. Es ersetzt jedoch alle Befehle, die Image Builder möglicherweise hinzugefügt hat, um sicherzustellen, dass Systems Manager installiert ist. Stellen

Sie bei der Verwendung sicher, dass der Systems Manager-Agent auf Ihrem Basis-Image vorinstalliert ist oder dass Sie die Installation in Ihre Benutzerdaten aufnehmen.

6. Im Abschnitt Komponenten müssen Sie mindestens eine Build-Komponente auswählen.

   Im Bereich Komponenten erstellen – Amazon Linux können Sie die auf der Seite aufgeführten Komponenten durchsuchen. Verwenden Sie die Paginierungssteuerung in der oberen rechten Ecke, um durch zusätzliche Komponenten zu navigieren, die für Ihr Basis-Image-Betriebssystem verfügbar sind. Sie können auch nach bestimmten Komponenten suchen oder Ihre eigene Build-Komponente mit dem Komponentenmanager erstellen.

   Wählen Sie für dieses Tutorial wie folgt eine Komponente aus, die Linux mit den neuesten Sicherheitsupdates aktualisiert:

   a. Filtern Sie die Ergebnisse, indem Sie das Wort `update` in die Suchleiste eingeben, die sich oben im Bereich befindet.

   b. Aktivieren Sie das Kontrollkästchen für die `update-linux` Build-Komponente.

   c. Scrollen Sie nach unten und wählen Sie in der oberen rechten Ecke der Liste Ausgewählte Komponenten die Option Alle erweitern aus.

   d. Behalten Sie die Standardeinstellung für Versioning-Optionen bei (Verwenden Sie die neueste verfügbare Komponentenversion).

   > **Note**
   >
   > Diese Einstellung stellt sicher, dass Ihre Pipeline semantische Versionsverwaltung für die ausgewählte Komponente verwendet, um Abhängigkeitsaktualisierungen für automatisch geplante Aufträge zu erkennen. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

   Wenn Sie eine Komponente mit Eingabeparametern ausgewählt haben, werden in diesem Bereich auch die Parameter angezeigt. Parameter werden in diesem Tutorial nicht behandelt. Weitere Informationen zur Verwendung von Eingabeparametern in Ihren Komponenten und zum Festlegen dieser Parameter in Ihren Rezepten finden Sie unter Verwalten von AWSTOE Komponentenparametern mit EC2 Image Builder.

   Komponenten neu anordnen (optional)

Wenn Sie mehr als eine Komponente ausgewählt haben, die in Ihr Image aufgenommen werden soll, können Sie die - drag-and-drop Aktion verwenden, um sie in der Reihenfolge neu anzuordnen, in der sie während des Build-Prozesses ausgeführt werden sollen.

> **Note**
>
> CIS-Hardening-Komponenten folgen nicht den Standardregeln für die Komponentenreihenfolge in Image-Builder-Rezepten. Die CIS-Harding-Komponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests mit Ihrem Ausgabe-Image ausgeführt werden.

1. Scrollen Sie zurück zur Liste der verfügbaren Komponenten.
2. Aktivieren Sie das Kontrollkästchen für die `update-linux-kernel-mainline` Build-Komponente (oder eine andere Komponente Ihrer Wahl).
3. Scrollen Sie nach unten zur Liste Ausgewählte Komponenten, um zu sehen, dass es mindestens zwei Ergebnisse gibt.
4. Neu hinzugefügte Komponenten haben möglicherweise nicht ihre Versioning- oder Eingabeparametereinstellungen erweitert. Um die Einstellungen für Versioning-Optionen oder Eingabeparameter zu erweitern, können Sie den Pfeil neben dem Namen der Einstellung auswählen. Um alle Einstellungen für alle ausgewählten Komponenten zu erweitern, können Sie die Option Alle erweitern deaktivieren und aktivieren.
5. Wählen Sie eine der Komponenten aus und ziehen Sie sie nach oben oder unten, um die Reihenfolge zu ändern, in der die Komponenten ausgeführt werden.
6. Um die `update-linux-kernel-mainline` Komponente zu entfernen, wählen Sie `X` aus der oberen rechten Ecke des Komponentenfelds aus.
7. Wiederholen Sie den vorherigen Schritt, um alle anderen Komponenten zu entfernen, die Sie möglicherweise hinzugefügt haben, wobei nur die `update-linux` Komponente ausgewählt bleibt.

7. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 3: Infrastrukturkonfiguration definieren – optional

Image Builder startet EC2-Instances in Ihrem Konto, um Images anzupassen und Validierungstests durchzuführen. Die Konfigurationseinstellungen für die Infrastruktur geben Infrastrukturdetails für die Instances an, die AWS-Konto während des Build-Prozesses in Ihrem ausgeführt werden.

Im Abschnitt Infrastrukturkonfiguration verwenden die Konfigurationsoptionen standardmäßig `Create infrastructure configuration using service defaults`. Dadurch werden eine IAM-Rolle und das zugehörige Instance-Profil für die EC2-Build- und Test-Instances erstellt, die zum Konfigurieren Ihres Images verwendet werden. Weitere Informationen zu Konfigurationseinstellungen für die Infrastruktur finden Sie unter CreateInfrastructureConfiguration in der EC2 Image Builder-API-Referenz.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

> **Note**
>
> Um ein Subnetz anzugeben, das für eine private VPC verwendet werden soll, können Sie Ihre eigene benutzerdefinierte Infrastrukturkonfiguration erstellen oder Einstellungen verwenden, die Sie bereits erstellt haben.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 4: Definieren von Verteilungseinstellungen – optional

Zu den Verteilungskonfigurationen gehören der Name des Ausgabe-AMI, spezifische Regionseinstellungen für die Verschlüsselung, Startberechtigungen und AWS-Konten, Organisationen und Organisationseinheiten (OUs), die das Ausgabe-AMI starten können, und Lizenzkonfigurationen.

Im Abschnitt Verteilungseinstellungen verwenden die Konfigurationsoptionen standardmäßig `Create distribution settings using service defaults`. Diese Option verteilt das Ausgabe-AMI an die aktuelle Region. Weitere Informationen zum Konfigurieren Ihrer Verteilungseinstellungen finden Sie unter Verwalten von EC2 Image Builder-Verteilungseinstellungen.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 5: Prüfen

Im Abschnitt Überprüfen werden alle von Ihnen konfigurierten Einstellungen angezeigt. Um Informationen in einem bestimmten Abschnitt zu bearbeiten, wählen Sie die Schaltfläche Bearbeiten in der oberen rechten Ecke des Schrittabschnitts. Wenn Sie beispielsweise Ihren Pipeline-Namen ändern möchten, wählen Sie die Schaltfläche Bearbeiten in der oberen rechten Ecke des Abschnitts Schritt 1: Pipeline-Details.

1. Wenn Sie Ihre Einstellungen überprüft haben, wählen Sie Pipeline erstellen, um Ihre Pipeline zu erstellen.
2. Oben auf der Seite werden Erfolgs- oder Fehlermeldungen angezeigt, da Ihre Ressourcen für Verteilungseinstellungen, Infrastrukturkonfiguration, Ihr neues Rezept und die Pipeline erstellt werden. Um Details für eine Ressource anzuzeigen, einschließlich der Ressourcen-ID, wählen Sie Details anzeigen aus.
3. Nachdem Sie sich die Details für eine Ressource angesehen haben, können Sie Details zu anderen Ressourcen anzeigen lassen, indem Sie den Ressourcentyp im Navigationsbereich auswählen. Um beispielsweise Details zu Ihrer neuen Pipeline anzuzeigen, wählen Sie im Navigationsbereich Image-Pipelines aus. Wenn Ihr Build erfolgreich war, wird Ihre neue Pipeline in der Liste Image-Pipelines angezeigt.

Schritt 6: Bereinigen

Ihre Image-Builder-Umgebung benötigt wie Ihr Heim regelmäßige Wartungsarbeiten, um Ihnen zu helfen, das benötigte zu finden und Ihre Aufgaben auszuführen, ohne dass Sie über eine Überlastung verfügen. Bereinigen Sie regelmäßig temporäre Ressourcen, die Sie zum Testen erstellt haben. Andernfalls vergessen Sie möglicherweise diese Ressourcen, und speichern sich später nicht, wofür sie verwendet wurden. Zu diesem Zeitpunkt ist es möglicherweise nicht klar, ob Sie sie sicher loslegen können.

> **Tip**
>
> Um Abhängigkeitsfehler beim Löschen von Ressourcen zu vermeiden, stellen Sie sicher, dass Sie Ihre Ressourcen in der folgenden Reihenfolge löschen:
>
> 1. Image-Pipeline
> 2. Image-Rezept

3. Alle verbleibenden Ressourcen

Gehen Sie folgendermaßen vor, um die Ressourcen zu bereinigen, die Sie für dieses Tutorial erstellt haben:

Löschen der Pipeline

1. Um eine Liste der Build-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.
2. Aktivieren Sie das Kontrollkästchen neben Pipeline-Name, um die Pipeline auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Pipelines im Menü Aktionen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen des Rezepts

1. Um eine Liste der unter Ihrem Konto erstellten Rezepte anzuzeigen, wählen Sie im Navigationsbereich Image-Rezepte aus.
2. Aktivieren Sie das Kontrollkästchen neben Rezeptname, um das Rezept auszuwählen, das Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Rezepte im Menü Aktionen die Option Rezept löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Infrastrukturkonfiguration löschen

1. Um eine Liste der Infrastrukturkonfigurationen anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Infrastrukturkonfiguration auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Infrastrukturkonfigurationen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen von Verteilungseinstellungen

1. Um eine Liste der unter Ihrem Konto erstellten Verteilungseinstellungen anzuzeigen, wählen Sie im Navigationsbereich Verteilungseinstellungen aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Verteilungseinstellungen auszuwählen, die Sie für dieses Tutorial erstellt haben.
3. Wählen Sie oben im Bereich Verteilungseinstellungen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen des Images

Gehen Sie wie folgt vor, um zu überprüfen, ob Sie alle Bilder gelöscht haben, die aus der Tutorial-Pipeline erstellt wurden. In diesem Tutorial wird wahrscheinlich kein Image erstellt, es sei denn, es ist genügend Zeit seit der Erstellung der Pipeline, die es ausführt, gemäß dem Build-Zeitplan verstrichen.

1. Um eine Liste der unter Ihrem Konto erstellten Images anzuzeigen, wählen Sie im Navigationsbereich Images aus.
2. Wählen Sie die Image-Version für das Image aus, das Sie entfernen möchten. Dadurch wird die Seite Image-Build-Versionen geöffnet.
3. Aktivieren Sie das Kontrollkästchen neben Version für jedes Image, das Sie löschen möchten. Sie können mehr als eine Image-Version gleichzeitig auswählen.
4. Wählen Sie oben im Bereich Image-Build-Versionen die Option Version löschen aus.
5. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Erstellen einer Container-Image-Pipeline mit dem EC2 Image Builder-Konsolenassistenten

Dieses Tutorial führt Sie durch das Erstellen einer automatisierten Pipeline zum Erstellen und Verwalten eines benutzerdefinierten EC2 Image Builder Docker-Images mit dem Assistenten zum Erstellen einer Image-Pipeline. Damit Sie die Schritte effizient ausführen können, werden Standardeinstellungen verwendet, wenn sie verfügbar sind, und optionale Abschnitte werden übersprungen.

Erstellen eines Image-Pipeline-Workflows

Schritt 1: Festlegen von Pipeline-Details

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um mit der Erstellung Ihrer Pipeline zu beginnen, wählen Sie Image-Pipeline erstellen aus.
3. Geben Sie im Abschnitt Allgemein Ihren Pipeline-Namen ein (erforderlich).
4. Im Abschnitt Zeitplan erstellen können Sie die Standardwerte für die Zeitplanoptionen beibehalten. Beachten Sie, dass die für den Standardzeitplan angezeigte Zeitzone die Universal Coordinated Time (UTC) ist. Weitere Informationen zur UTC-Zeit und zum Ermitteln des Offsets für Ihre Zeitzone finden Sie unter Zeitzonenabkürzungen – Weltweite Liste.

   Wählen Sie für Einstellungen für Abhängigkeitsaktualisierungen die Option Pipeline zum geplanten Zeitpunkt ausführen aus, wenn es Abhängigkeitsaktualisierungen gibt. Diese Einstellung führt dazu, dass Ihre Pipeline vor dem Start des Builds nach Updates sucht. Wenn es keine Updates gibt, wird der geplante Pipeline-Build übersprungen.

   > **Note**
   >
   > Um sicherzustellen, dass Ihre Pipeline Abhängigkeitsaktualisierungen erkennt und wie erwartet erstellt, müssen Sie semantische Versionsverwaltung (x.x.x) für Ihr Basis-Image und Ihre Komponenten verwenden. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

5. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 2: Rezept auswählen

1. Image Builder verwendet standardmäßig das vorhandene Rezept im Abschnitt Rezept. Wählen Sie zum ersten Mal die Option Neues Rezept erstellen aus.
2. Wählen Sie im Abschnitt Image-Typ die Option Docker-Image aus, um eine Container-Pipeline zu erstellen, die ein Docker-Image erzeugt und an Amazon-ECR-Repositorys in Zielregionen verteilt.
3. Geben Sie im Abschnitt Allgemein die folgenden Pflichtfelder ein:

   - Name – Ihr Rezeptname
   - Version – Ihre Rezeptversion (verwenden Sie das Format <major>.<minor>.<patch>, wobei major, minor und patch Ganzzahlwerte sind). Neue Rezepte beginnen im Allgemeinen mit `1.0.0`.
4. Behalten Sie im Abschnitt Quellbild die Standardwerte für Image auswählen, Image-Betriebssystem (OS) und Image-Ursprung bei. Dies führt zu einer Liste von Amazon Linux 2-Container-Images, die von Amazon verwaltet werden und aus denen Sie für Ihr Basis-Image wählen können.

   a. Wählen Sie in der Dropdownliste Image name ein Image aus.

   b. Behalten Sie die Standardeinstellung für Optionen für die automatische Versionsverwaltung bei (Verwenden Sie die neueste verfügbare Betriebssystemversion).

   > **Note**
   >
   > Diese Einstellung stellt sicher, dass Ihre Pipeline semantische Versionsverwaltung für das Basis-Image verwendet, um Abhängigkeitsaktualisierungen für automatisch geplante Aufträge zu erkennen. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.
5. Im Abschnitt Komponenten müssen Sie mindestens eine Build-Komponente auswählen.

   Im Bereich Komponenten erstellen – Amazon Linux können Sie die auf der Seite aufgeführten Komponenten durchsuchen. Verwenden Sie die Paginierungssteuerung in der oberen rechten Ecke, um durch zusätzliche Komponenten zu navigieren, die für Ihr Basis-Image-Betriebssystem verfügbar sind. Sie können auch nach bestimmten Komponenten suchen oder Ihre eigene Build-Komponente mit dem Komponentenmanager erstellen.

Wählen Sie für dieses Tutorial wie folgt eine Komponente aus, die Linux mit den neuesten Sicherheitsupdates aktualisiert:

a. Filtern Sie die Ergebnisse, indem Sie das Wort `update` in die Suchleiste eingeben, die sich oben im Bereich befindet.

b. Aktivieren Sie das Kontrollkästchen für die `update-linux` Build-Komponente.

c. Scrollen Sie nach unten und wählen Sie in der oberen rechten Ecke der Liste Ausgewählte Komponenten die Option Alle erweitern aus.

d. Behalten Sie die Standardeinstellung für Versioning-Optionen bei (Verwenden Sie die neueste verfügbare Komponentenversion).

> **Note**
>
> Diese Einstellung stellt sicher, dass Ihre Pipeline semantische Versionsverwaltung für die ausgewählte Komponente verwendet, um Abhängigkeitsaktualisierungen für automatisch geplante Aufträge zu erkennen. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

Wenn Sie eine Komponente mit Eingabeparametern ausgewählt hätten, würden Sie auch die Parameter in diesem Bereich sehen. Parameter werden in diesem Tutorial nicht behandelt. Weitere Informationen zur Verwendung von Eingabeparametern in Ihren Komponenten und zum Festlegen dieser Parameter in Ihren Rezepten finden Sie unter Verwalten von AWSTOE Komponentenparametern mit EC2 Image Builder.

Komponenten neu anordnen (optional)

Wenn Sie mehr als eine Komponente ausgewählt haben, die in Ihr Image aufgenommen werden soll, können Sie die - drag-and-drop Aktion verwenden, um sie in der Reihenfolge neu anzuordnen, in der sie während des Build-Prozesses ausgeführt werden sollen.

> **Note**
>
> CIS-Hardening-Komponenten folgen nicht den Standardregeln für die Komponentenreihenfolge in Image-Builder-Rezepten. Die CIS-Harding-Komponenten

werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests mit Ihrem Ausgabe-Image ausgeführt werden.

1. Scrollen Sie zurück zur Liste der verfügbaren Komponenten.
2. Aktivieren Sie das Kontrollkästchen für die `update-linux-kernel-mainline` Build-Komponente (oder eine andere Komponente Ihrer Wahl).
3. Scrollen Sie nach unten zur Liste Ausgewählte Komponenten, um zu sehen, dass mindestens zwei Ergebnisse vorliegen.
4. Neu hinzugefügte Komponenten haben möglicherweise ihre Versionsverwaltung nicht erweitert. Um die Versioning-Optionen zu erweitern, können Sie entweder den Pfeil neben Versioning-Optionen auswählen oder die Option Alle erweitern aus- und aktivieren, um die Versioning für alle ausgewählten Komponenten zu erweitern.
5. Wählen Sie eine der Komponenten aus und ziehen Sie sie nach oben oder unten, um die Reihenfolge zu ändern, in der die Komponenten ausgeführt werden.
6. Um die `update-linux-kernel-mainline` Komponente zu entfernen, wählen Sie `X` aus der oberen rechten Ecke des Komponentenfelds aus.
7. Wiederholen Sie den vorherigen Schritt, um alle anderen Komponenten zu entfernen, die Sie möglicherweise hinzugefügt haben, wobei nur die `update-linux` Komponente ausgewählt bleibt.

6. Wählen Sie im Abschnitt Dockerfile-Vorlage die Option Beispiel verwenden aus. Beachten Sie im Bereich Inhalt die Kontextvariablen, in denen Image Builder Build-Informationen oder Skripts basierend auf Ihrem Container-Image-Rezept ablegt.
7. Geben Sie im Abschnitt Ziel-Repository den Namen des Amazon-ECR-Repositorys an, das Sie als Voraussetzung für dieses Tutorial erstellt haben. Dieses Repository wird als Standardeinstellung für die Verteilungskonfiguration in der Region verwendet, in der die Pipeline ausgeführt wird (Region 1).

> **Note**
>
> Das Ziel-Repository muss vor der Verteilung in Amazon ECR für alle Zielregionen vorhanden sein.

8. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 3: Infrastrukturkonfiguration definieren – optional

Image Builder startet EC2-Instances in Ihrem Konto, um Images anzupassen und Validierungstests durchzuführen. Die Konfigurationseinstellungen für die Infrastruktur geben Infrastrukturdetails für die Instances an, die AWS-Konto während des Build-Prozesses in Ihrem ausgeführt werden.

Im Abschnitt Infrastrukturkonfiguration verwenden die Konfigurationsoptionen standardmäßig `Create infrastructure configuration using service defaults`. Dadurch werden eine IAM-Rolle und das zugehörige Instance-Profil erstellt, die von Build-Instances zur Konfiguration Ihrer Container-Images verwendet werden. Sie können auch Ihre eigene benutzerdefinierte Infrastrukturkonfiguration erstellen oder Einstellungen verwenden, die Sie bereits erstellt haben. Weitere Informationen zu Konfigurationseinstellungen für die Infrastruktur finden Sie unter CreateInfrastructureConfiguration in der EC2 Image Builder API-Referenz.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 4: Definieren von Verteilungseinstellungen – optional

Verteilungseinstellungen bestehen aus den Zielregionen und dem Namen des Amazon-ECR-Ziel-Repositorys. Ausgabe-Docker-Images werden im benannten Amazon ECR-Repository in jeder Region bereitgestellt.

Im Abschnitt Verteilungseinstellungen verwenden die Konfigurationsoptionen standardmäßig `Create distribution settings using service defaults`. Mit dieser Option wird das Ausgabe-Docker-Image an das Amazon-ECR-Repository verteilt, das in Ihrem Container-Rezept für die Region angegeben ist, in der Ihre Pipeline ausgeführt wird (Region 1). Wenn Sie auswählen`Create new distribution settings`, können Sie das ECR-Repository für die aktuelle Region überschreiben und weitere Regionen für die Verteilung hinzufügen.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 5: Prüfen

Im Abschnitt Überprüfen werden alle von Ihnen konfigurierten Einstellungen angezeigt. Um Informationen in einem bestimmten Abschnitt zu bearbeiten, wählen Sie die Schaltfläche Bearbeiten

in der oberen rechten Ecke des Schrittabschnitts. Wenn Sie beispielsweise Ihren Pipeline-Namen ändern möchten, wählen Sie die Schaltfläche Bearbeiten in der oberen rechten Ecke des Abschnitts Schritt 1: Pipeline-Details.

1. Wenn Sie Ihre Einstellungen überprüft haben, wählen Sie Pipeline erstellen, um Ihre Pipeline zu erstellen.
2. Oben auf der Seite werden Erfolgs- oder Fehlermeldungen angezeigt, da Ihre Ressourcen für Verteilungseinstellungen, Infrastrukturkonfiguration, Ihr neues Rezept und die Pipeline erstellt werden. Um Details für eine Ressource anzuzeigen, einschließlich der Ressourcen-ID, wählen Sie Details anzeigen aus.
3. Nachdem Sie sich die Details für eine Ressource angesehen haben, können Sie Details zu anderen Ressourcen anzeigen lassen, indem Sie den Ressourcentyp im Navigationsbereich auswählen. Um beispielsweise Details zu Ihrer neuen Pipeline anzuzeigen, wählen Sie im Navigationsbereich Image-Pipelines aus. Wenn Ihr Build erfolgreich war, wird Ihre neue Pipeline in der Liste Image-Pipelines angezeigt.

Schritt 6: Bereinigen

Ihre Image-Builder-Umgebung benötigt wie Ihr Heim regelmäßige Wartungsarbeiten, um Ihnen zu helfen, das benötigte zu finden und Ihre Aufgaben auszuführen, ohne dass Sie über eine Überlastung verfügen. Bereinigen Sie regelmäßig temporäre Ressourcen, die Sie zum Testen erstellt haben. Andernfalls vergessen Sie möglicherweise diese Ressourcen, und speichern sich später nicht, wofür sie verwendet wurden. Zu diesem Zeitpunkt ist es möglicherweise nicht klar, ob Sie sie sicher loslegen können.

> **Tip**
>
> Um Abhängigkeitsfehler beim Löschen von Ressourcen zu vermeiden, stellen Sie sicher, dass Sie Ihre Ressourcen in der folgenden Reihenfolge löschen:
>
> 1. Image-Pipeline
> 2. Image-Rezept
> 3. Alle verbleibenden Ressourcen

Gehen Sie wie folgt vor, um die Ressourcen zu bereinigen, die Sie für dieses Tutorial erstellt haben:

Löschen der Pipeline

1. Um eine Liste der Build-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.
2. Aktivieren Sie das Kontrollkästchen neben Pipeline-Name, um die Pipeline auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Pipelines im Menü Aktionen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen des Container-Rezepts

1. Um eine Liste der Container-Rezepte anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Container-Rezepte aus.
2. Aktivieren Sie das Kontrollkästchen neben Rezeptname, um das Rezept auszuwählen, das Sie löschen möchten.
3. Wählen Sie oben im Bereich Container-Rezepte im Menü Aktionen die Option Rezept löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Infrastrukturkonfiguration löschen

1. Um eine Liste der Infrastrukturkonfigurationen anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Infrastrukturkonfiguration auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Infrastrukturkonfigurationen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen von Verteilungseinstellungen

1. Um eine Liste der unter Ihrem Konto erstellten Verteilungseinstellungen anzuzeigen, wählen Sie im Navigationsbereich Verteilungseinstellungen aus.

2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Verteilungseinstellungen auszuwählen, die Sie für dieses Tutorial erstellt haben.
3. Wählen Sie oben im Bereich Verteilungseinstellungen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen des Images

Gehen Sie wie folgt vor, um zu überprüfen, ob Sie alle Bilder gelöscht haben, die aus der Tutorial-Pipeline erstellt wurden. In diesem Tutorial wird wahrscheinlich kein Image erstellt, es sei denn, es ist genügend Zeit seit der Erstellung Ihrer Pipeline, die es ausführt, gemäß dem Build-Zeitplan verstrichen.

1. Um eine Liste der unter Ihrem Konto erstellten Images anzuzeigen, wählen Sie im Navigationsbereich Images aus.
2. Wählen Sie die Image-Version für das Image aus, das Sie entfernen möchten. Dadurch wird die Seite Image-Build-Versionen geöffnet.
3. Aktivieren Sie das Kontrollkästchen neben Version für jedes Image, das Sie löschen möchten. Sie können mehr als eine Image-Version gleichzeitig auswählen.
4. Wählen Sie oben im Bereich Image-Build-Versionen die Option Version löschen aus.
5. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

AWS Task Orchestrator and Executor Komponentenmanager

EC2 Image Builder verwendet die AWS Task Orchestrator and Executor (AWSTOE)-Anwendung, um komplexe Workflows zu orchestrieren, Systemkonfigurationen zu ändern und Ihre Systeme zu testen, ohne Code zu schreiben. Diese Anwendung verwaltet und führt Komponenten aus, die ihr deklaratives Dokumentschema implementieren.

Da es sich um eine eigenständige Anwendung handelt, ist keine zusätzliche Servereinrichtung erforderlich. Es kann in jeder Cloud-Infrastruktur und On-Premises ausgeführt werden.

Inhalt

AWSTOE-Downloads

Um zu installierenAWSTOE, wählen Sie den Download-Link für Ihre Architektur und Plattform. Wenn Sie an einen VPC-Endpunkt für Ihren Service anfügen (z. B. Image Builder), muss eine benutzerdefinierte Endpunktrichtlinie angehängt sein, die den Zugriff auf den S3-Bucket zum AWSTOE Herunterladen enthält. Andernfalls können Ihre Build- und Test-Instances das Bootstrap-Skript (`bootstrap.sh`) nicht herunterladen und die AWSTOE Anwendung installieren. Weitere Informationen finden Sie unter Erstellen einer VPC-Endpunktrichtlinie für Image Builder.

Important

AWS stellt die Unterstützung für die TLS-Versionen 1.0 und 1.1 ein. Um zum AWSTOE Herunterladen auf den S3-Bucket zuzugreifen, muss Ihre Client-Software TLS Version 1.2 oder höher verwenden. Weitere Informationen finden Sie in diesem AWS Security Blog-Beitrag.

Architektur	Plattform	Download-Link	Beispiel
386	AL 2 und 2023 RHEL 7 und 8 Ubuntu 16.04, 18.04, 20.04 und 22.04 CentOS 7 und 8 SUSE 12 und 15	`https://awstoe- <region>.s3.com/latest/linux/386/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/386/awstoe
AMD64	Windows Server 2012 R2, 2016, 2019 und 2022	`https://awstoe- <region>.s3.com/latest/windows/amd64/awstoe.exe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/windows/amd64/awstoe.exe
AMD64	AL 2 und 2023 RHEL 7 und 8 Ubuntu 16.04, 18.04, 20.04 und 22.04 CentOS 7 und 8 CentOS Stream 8 SUSE 12 und 15	`https://awstoe- <region>.s3.com/latest/linux/amd64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/awstoe

Architektur	Plattform	Download-Link	Beispiel
ARM64	AL 2 und 2023 RHEL 7 und 8 Ubuntu 16.04, 18.04, 20.04 und 22.04 CentOS 7 und 8 CentOS Stream 8 SUSE 12 und 15	`https://awstoe- <region>.s3.s3.com/latest/linux/arm64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/arm64/awstoe

Unterstützte Regionen

AWSTOE wird als eigenständige Anwendung in den folgenden Regionen unterstützt.

AWS-Region-Name	AWS-Region
USA Ost (Ohio)	us-east-2
US East (N. Virginia)	us-east-1
AWS GovCloud (USA-Ost)	us-gov-east-1
AWS GovCloud (USA-West)	us-gov-west-1
USA West (Nordkalifornien)	us-west-1
US West (Oregon)	us-west-2
Afrika (Kapstadt)	af-south-1
Asien-Pazifik (Hongkong)	ap-east-1
Asien-Pazifik (Osaka)	ap-northeast-3
Asien-Pazifik (Seoul)	ap-northeast-2

AWS-Region-Name	AWS-Region
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Hyderabad)	ap-south-2
Asien-Pazifik (Singapur)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asien-Pazifik (Jakarta)	ap-southeast-3
Asien-Pazifik (Tokio)	ap-northeast-1
Canada (Central)	ca-central-1
Europa (Frankfurt)	eu-central-1
Europa (Zürich)	eu-central-2
Europa (Stockholm)	eu-north-1
Europa (Mailand)	eu-south-1
Europa (Spanien)	eu-south-2
Europa (Irland)	eu-west-1
Europe (London)	eu-west-2
Europe (Paris)	eu-west-3
Israel (Tel Aviv)	il-central-1
Naher Osten (VAE)	me-central-1
Naher Osten (Bahrain)	me-south-1
South America (São Paulo)	sa-east-1
China (Beijing)	cn-north-1

AWS-Region-Name	AWS-Region
China (Ningxia)	cn-northwest-1

Erste Schritte mit AWSTOE

Die AWS Task Orchestrator and Executor (AWSTOE)-Anwendung ist eine eigenständige Anwendung, die Befehle innerhalb eines Komponentendefinitions-Frameworks erstellt, validiert und ausführt. -AWSServices können verwenden, AWSTOE um Workflows zu orchestrieren, Software zu installieren, Systemkonfigurationen zu ändern und Image-Builds zu testen.

Gehen Sie wie folgt vor, um die AWSTOE Anwendung zu installieren und zum ersten Mal zu verwenden.

Überprüfen der Signatur des AWSTOE Installations-Downloads

In diesem Abschnitt wird der empfohlene Prozess zur Überprüfung der Gültigkeit des Installations-Downloads für AWSTOE auf Linux- und Windows-basierten Betriebssystemen beschrieben.

Themen

Überprüfen der Signatur des AWSTOE Installations-Downloads unter Linux

In diesem Thema wird der empfohlene Prozess zur Überprüfung der Gültigkeit des Installations-Downloads für die AWSTOE unter Linux-basierten Betriebssystemen beschrieben.

Wenn Sie eine Anwendung aus dem Internet herunterladen, empfehlen wir Ihnen, die Identität des Softwareherausgebers zu authentifizieren. Überprüfen Sie außerdem, ob die Anwendung seit ihrer Veröffentlichung nicht verändert oder beschädigt wurde. Dies schützt Sie davor, eine Version der Anwendung zu installieren, die einen Virus oder einen anderen bösartigen Code enthält.

Wenn Sie nach dem Ausführen der Schritte in diesem Thema feststellen, dass die Software für das verändert oder beschädigt AWSTOE wurde, führen Sie die Installationsdatei nicht aus. Wenden Sie sich stattdessen an AWS Support . Weitere Informationen zu Ihren Support-Optionen finden Sie unter AWS Support.

AWSTOE -Dateien für Linux-basierte Betriebssysteme werden mit signiertGnuPG, einer Open-Source-Implementierung des Pretty Good Privacy (OpenPGP)-Standards für sichere digitale Signaturen. GnuPG (auch bekannt als GPG) bietet Authentifizierung und Integritätsprüfung durch eine digitale Signatur. Amazon EC2 veröffentlicht einen öffentlichen Schlüssel und Signaturen, mit denen Sie die heruntergeladenen Amazon EC2-CLI-Tools überprüfen können. Weitere Informationen zu PGP und GnuPG (GPG) finden Sie unter http://www.gnupg.org.

Der erste Schritt besteht darin, eine Vertrauensstellung mit dem Software-Publisher zu schaffen. Laden Sie den öffentlichen Schlüssel des Softwareherausgebers herunter, überprüfen Sie, ob der Besitzer des öffentlichen Schlüssels derjenige ist, der er behauptet zu sein, und fügen Sie dann den öffentlichen Schlüssel zu Ihrem Schlüsselbund hinzu. Ihr Schlüsselbund ist eine Sammlung von bekannten öffentlichen Schlüsseln. Nachdem Sie die Echtheit des öffentlichen Schlüssels überprüft haben, können Sie ihn verwenden, um die Signatur der Anwendung zu überprüfen.

Themen

Installieren der GPG-Tools

Wenn Sie das Betriebssystem Linux oder Unix verwenden, sind die GPG-Tools wahrscheinlich bereits installiert. Um zu testen, ob die Tools auf Ihrem System installiert sind, geben Sie gpg in einer Eingabeaufforderung ein. Wenn die GPG-Tools installiert sind, sehen Sie eine Eingabeaufforderung. Wenn die GPG-Tools nicht installiert sind, wird eine Fehlermeldung angezeigt, die besagt, dass der Befehl nicht gefunden werden kann. Sie können das GnuPG-Paket von einem Repository aus installieren.

So installieren Sie GPG-Tools auf Debian-basiertem Linux

- Führen Sie von einem Terminal folgenden Befehl aus: apt-get install gnupg.

So installieren Sie GPG-Tools unter Red-Hat-basiertem Linux

- Führen Sie von einem Terminal folgenden Befehl aus: yum install gnupg.

Authentifizieren und Importieren des öffentlichen Schlüssels

Der nächste Schritt des Vorgangs besteht darin, den öffentlichen Schlüssel von AWSTOE zu authentifizieren und ihn als vertrauenswürdigen Schlüssel Ihrem GPG-Schlüsselbund hinzuzufügen.

So authentifizieren und importieren Sie den öffentlichen Schlüssel von AWSTOE

1. Besorgen Sie sich ein Exemplar unseres öffentlichen GPG-Schlüssels, indem Sie einen der folgenden Schritte ausführen:

   - Laden Sie den Schlüssel von https://awstoe-**<region>**.s3.**<region>**amazonaws.com/assets/awstoe.gpg. herunter Beispiel: https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/assets/awstoe.gpg
   - Kopieren Sie den Schlüssel aus dem folgenden Text und fügen Sie ihn in eine Datei namens `awstoe.gpg` ein. Vergewissern Sie sich, alles Folgende einzubeziehen:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQENBF8UqwsBCACdiRF2bkZYaFSDPFC+LIkWLwFvtUCRwAHtD8KIwTJ6LVn3fHAU
GhuK0ZH9mRrqRT2bq/xJjGsnF9VqTj2AJqndGJdDjz75YCZYM+ocZ+r5HSJaeW9i
S5dykHj7Txti2zHe0G5+W0v7v5bPi2sPHsN7XWQ7+G2AMEPTz8PjxY//I0DvMQns
Sle3l9hz6wCClz1l9LbBzTyHfSm5ucTXvNe88XX5Gmt37OCDM7vfli0Ctv8WFoLN
6jbxuA/sV71yIkPm9IYp3+GvaKeT870+sn8/JOOKE/U4sJV1ppbqmuUzDfhrZUaw
8eW8IN9A1FTIuWiZED/5L83UZuQs1S7s2PjlABEBAAG0GkFXU1RPRSA8YXdzdG9l
QGFtYXpvbi5jb20+iQE5BBMBCAAjBQJfFKsLAhsDBwsJCAcDAgEGFQgCCQoLBBYC
AwECHgECF4AACgkQ3r3BVvWuvFJGiwf9EVmrBR77+Qe/DUeXZJYoaFr7If/fVDZl
6V3TC6p0J0Veme7uXleRUTFOjzbh+7e5sDX19HrnPquzCnzfMiqbp4lSoeUuNdOf
FcpuTCQH+M+sIEIgPno4PLl0Uj2uE1o++mxmonBl/Krk+hly8hB2L/9n/vW3L7BN
OMb1Ll9PmgGPbWipcT8KRdz4SUex9TXGYzjlWb3jU3uXetdaQY1M3kVKE1siRsRN
YYDtpcjmwbhjpu4xm19aFqNoAHCDctEsXJA/mkU3erwIRocPyjAZE2dnlkL9ZkFZ
z9DQkcIarbCnybDM5lemBbdhXJ6hezJE/b17VA0t1fY04MoEkn6oJg==
=oyze
-----END PGP PUBLIC KEY BLOCK-----
```

2. Verwenden Sie an einer Eingabeaufforderung in dem Verzeichnis, in dem Sie awstoe.gpg gespeichert haben, den folgenden Befehl, um den AWSTOE öffentlichen Schlüssel in Ihren Schlüsselbund zu importieren.

```
gpg --import awstoe.gpg
```

Der Befehl gibt Ergebnisse wie die folgenden zurück:

```
gpg: key F5AEBC52: public key "AWSTOE <awstoe@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

Notieren Sie sich den Schlüsselwert. Sie brauchen ihn im nächsten Schritt. Im vorangegangenen Beispiel ist der Schlüsselwert `F5AEBC52`.

3. Überprüfen Sie den Fingerabdruck, indem Sie den folgenden Befehl ausführen und Schlüssel-Wert durch den Wert des vorherigen Schritts ersetzen:

```
gpg --fingerprint key-value
```

Dieser Befehl gibt Ergebnisse wie die folgenden zurück:

```
pub   2048R/F5AEBC52 2020-07-19
      Key fingerprint = F6DD E01C 869F D639 15E5  5742 DEBD C156 F5AE BC52
uid       [ unknown] AWSTOE <awstoe@amazon.com>
```

Zusätzlich sollte der Fingerabdruck-String identisch mit `F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52` sein, wie im voranstehenden Beispiel angezeigt. Vergleichen Sie den Schlüssel-Fingerabdruck mit demjenigen, der auf dieser Seite veröffentlicht ist.
Sie sollten übereinstimmen. Wenn sie nicht übereinstimmen, installieren Sie das AWSTOE Installationsskript nicht und wenden Sie sich an AWS Support.

Verifizieren der Signatur des Pakets

Nachdem Sie die GPG-Tools installiert, den öffentlichen Schlüssel für AWSTOE authentifiziert und importiert und überprüfen haben, dass der öffentliche Schlüssel vertrauenswürdig ist, sind Sie bereit, die Signatur des Installationsskripts zu überprüfen.

So überprüfen Sie die Signatur des -Installationsskripts

1. Führen Sie an einer Eingabeaufforderung den folgenden Befehl aus, um die Anwendungsbinärdatei herunterzuladen:

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe
```

Beispielsweise:

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe
```

Unterstützte Werte für **`architecture`** können `amd64` `386`, und sein `arm64`.

2. Führen Sie an einer Eingabeaufforderung den folgenden Befehl aus, um die Signaturdatei für die entsprechende Anwendungsbinärdatei aus demselben S3-Schlüsselpräfixpfad herunterzuladen:

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe.sig
```

Beispielsweise:

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe.sig
```

Unterstützte Werte für **`architecture`** können `amd64` `386`, und sein `arm64`.

3. Überprüfen Sie die Signatur, indem Sie den folgenden Befehl an einer Eingabeaufforderung in dem Verzeichnis ausführen, in dem Sie `awstoe.sig` und die AWSTOE-Installationsdatei gespeichert haben. Beide Dateien müssen vorhanden sein.

```
gpg --verify ./awstoe.sig ~/awstoe
```

Die Ausgabe sollte wie folgt aussehen:

```
gpg: Signature made Mon 20 Jul 2020 08:54:55 AM IST using RSA key ID F5AEBC52
gpg: Good signature from "AWSTOE awstoe@amazon.com" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52
```

Wenn die Ausgabe den Begriff `Good signature from "AWSTOE <awstoe@amazon.com>"` enthält, bedeutet dies, dass die Signatur erfolgreich überprüft wurde und Sie mit der Ausführung des AWSTOE-Installationsskripts fortfahren können.

Wenn die Ausgabe die Bezeichnung `BAD signature`enthält, überprüfen Sie, ob Sie das Verfahren korrekt durchgeführt haben. Wenn Sie diese Antwort weiterhin erhalten, führen Sie nicht die Installationsdatei aus, die Sie zuvor heruntergeladen haben, und wenden Sie sich an AWS Support.

Im Folgenden finden Sie Details zu den Warnungen, die möglicherweise angezeigt werden:

- WARNUNG: Dieser Schlüssel ist nicht mit einer vertrauenswürdigen Signatur zertifiziert! Es gibt keinen Hinweis darauf, dass die Signatur dem Eigentümer gehört. Idealerweise würden Sie ein - AWS Büro besuchen und den Schlüssel persönlich erhalten. Sie würden sie jedoch höchstwahrscheinlich von einer Website herunterladen. In diesem Fall handelt es sich bei der Website um eine AWS-Website.
- gpg: Keine endgültig vertrauenswürdigen Schlüssel gefunden Das bedeutet, dass der spezifische Schlüssel nicht von Ihnen oder von anderen Personen, denen Sie vertrauen, „letztläufig vertrauenswürdig“ ist.

Weitere Informationen finden Sie unter http://www.gnupg.org.

Überprüfen der Signatur des AWSTOE Installations-Downloads unter Windows

In diesem Thema wird der empfohlene Prozess zur Überprüfung der Gültigkeit der Installationsdatei für die AWS Task Orchestrator and Executor Anwendung unter Windows-basierten Betriebssystemen beschrieben.

Wenn Sie eine Anwendung aus dem Internet herunterladen, empfehlen wir Ihnen, die Identität des Softwareverlegers zu authentifizieren und zu überprüfen, ob die Anwendung seit ihrer Veröffentlichung nicht verändert oder beschädigt wurde. Dies schützt Sie davor, eine Version der Anwendung zu installieren, die einen Virus oder einen anderen bösartigen Code enthält.

Wenn Sie nach dem Ausführen der Schritte in diesem Thema feststellen, dass die Software für die AWSTOE Anwendung verändert oder beschädigt wurde, führen Sie die Installationsdatei nicht aus. Wenden Sie sich stattdessen an AWS Support.

Um die Gültigkeit der heruntergeladenen awstoe-Binärdatei auf Windows-basierten Betriebssystemen zu überprüfen, stellen Sie sicher, dass der Thumbprint seines Amazon-Services-Travel-Signaturzertifikats diesem Wert entspricht:

5B 77 F4 F0 C3 7A 8B 89 D9 A7 8F 54 B6 85 11 CE 9E A3 BF 17

> **Note**
>
> Derzeit führen wir eine neue Binärdatei ein. Wenn Ihr Ausstellerzertifikat nicht mit dem neuen Thumbprint übereinstimmt, überprüfen Sie, ob der Thumbprint-Wert wie folgt lautet:
> META B6 AC 0D 24 E6 90 1D 35 9D 69 38 4C DF 31 04 A7 3E 91 7A

Um diesen Wert zu überprüfen, gehen Sie wie folgt vor:

1. Klicken Sie mit der rechten Maustaste auf die heruntergeladen `awstoe.exe`, und öffnen Sie das Eigenschaften-Fenster.
2. Wählen Sie die Registerkarte Digital Signatures aus.
3. Wählen Sie in der Signature List die Option Amazon Services LLC und dann Details aus.
4. Falls die Registerkarte General nicht bereits ausgewählt ist, klicken Sie darauf und dann auf View Certificate.
5. Wählen Sie die Registerkarte Details aus, und anschließend die Option All (Alle) in der Dropdown-Liste Show (Zeigen), wenn diese nicht bereits ausgewählt ist.
6. Scrollen Sie nach unten zum Feld Thumbprint und wählen Sie Thumbprint aus. Der gesamte Thumbprint-Wert wird im unteren Fenster angezeigt.

   - Wenn der Thumbprint-Wert im unteren Fenster mit folgendem Wert identisch ist:

     5B 77 F4 F0 C3 7A 8B 89 D9 A7 8F 54 B6 85 11 CE 9E A3 BF 17

     dann ist Ihre heruntergeladene AWSTOE Binärdatei echt und kann sicher installiert werden.

     > **Note**
     >
     > Derzeit führen wir eine neue Binärdatei ein. Wenn Ihr Ausstellerzertifikat nicht mit dem neuen Thumbprint übereinstimmt, überprüfen Sie, ob der Thumbprint-Wert wie folgt lautet:
     > META B6 AC 0D 24 E6 90 1D 35 9D 69 38 4C DF 31 04 A7 3E 91 7A

- Wenn der Thumbprint-Wert im unteren Detailfenster nicht mit dem vorherigen Wert identisch ist, führen Sie nicht aus`awstoe.exe`.

Erste Schritte

- Schritt 1: Installieren von AWSTOE
- Schritt 2: Festlegen von AWS Anmeldeinformationen
- Schritt 3: Lokales Entwickeln von Komponentendokumenten
- Schritt 4: Validieren von AWSTOE Komponenten
- Schritt 5: Ausführen von AWSTOE Komponenten

Schritt 1: Installieren von AWSTOE

Um Komponenten lokal zu entwickeln, laden Sie die AWSTOE Anwendung herunter und installieren Sie sie.

1. Herunterladen der AWSTOE Anwendung

   Um zu installierenAWSTOE, wählen Sie den entsprechenden Download-Link für Ihre Architektur und Plattform aus. Eine vollständige Liste der Anwendungs-Download-Links finden Sie unter . AWSTOE-Downloads

   > **Important**
   >
   > AWS stellt die Unterstützung für die TLS-Versionen 1.0 und 1.1 ein. Um zum AWSTOE Herunterladen auf den S3-Bucket zuzugreifen, muss Ihre Client-Software TLS Version 1.2 oder höher verwenden. Weitere Informationen finden Sie in diesem AWS Security Blog-Beitrag.

2. Überprüfen der Signatur

   Die Schritte zum Überprüfen Ihres Downloads hängen von der Serverplattform ab, auf der Sie die AWSTOE Anwendung nach der Installation ausführen. Informationen zum Überprüfen Ihres Downloads auf einem Linux-Server finden Sie unter Überprüfen der Signatur unter Linux. Informationen zum Überprüfen Ihres Downloads auf einem Windows-Server finden Sie unter Überprüfen der Signatur unter Windows.

> **Important**
>
> AWSTOE wird direkt von seinem Download-Speicherort aus aufgerufen. Es ist kein separater Installationsschritt erforderlich. Dies bedeutet auch, dass Änderungen an der lokalen Umgebung vornehmen AWSTOE kann.
> Um sicherzustellen, dass Sie Änderungen während der Komponentenentwicklung isolieren, empfehlen wir Ihnen, eine EC2-Instance zum Entwickeln und Testen von AWSTOE Komponenten zu verwenden.

Schritt 2: Festlegen von AWS Anmeldeinformationen

AWSTOE erfordert AWS Anmeldeinformationen, um eine Verbindung zu anderen herzustellenAWS-Services, z. B. Amazon S3 und Amazon CloudWatch, wenn Aufgaben ausgeführt werden, z. B.:

- Herunterladen von AWSTOE Dokumenten von einem vom Benutzer bereitgestellten Amazon S3-Pfad.
- Ausführen von - `S3Download` oder -`S3Upload`Aktionsmodulen.
- Streamen von Protokollen an CloudWatch, wenn aktiviert.

Wenn Sie AWSTOE auf einer EC2-Instance ausführen, AWSTOE verwendet das Ausführen von die gleichen Berechtigungen wie die IAM-Rolle, die der EC2-Instance zugeordnet ist.

Weitere Informationen zu IAM-Rollen für EC2 finden Sie unter IAM-Rollen für Amazon EC2.

Die folgenden Beispiele zeigen, wie Sie AWS Anmeldeinformationen mithilfe der `AWS_SECRET_ACCESS_KEY` Umgebungsvariablen `AWS_ACCESS_KEY_ID` und festlegen.

Um diese Variablen unter Linux, macOS oder Unix festzulegen, verwenden Sie `export`.

```
$ export AWS_ACCESS_KEY_ID=your_access_key_id
```

```
$ export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Um diese Variablen unter Windows mit festzulegen PowerShell, verwenden Sie `$env`.

```
C:\> $env:AWS_ACCESS_KEY_ID=your_access_key_id
```

```
C:\> $env:AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Um diese Variablen unter Windows über die Eingabeaufforderung festzulegen, verwenden Sie `set`.

```
C:\> set AWS_ACCESS_KEY_ID=your_access_key_id
```

```
C:\> set AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Schritt 3: Lokales Entwickeln von Komponentendokumenten

AWSTOE -Komponenten werden mit Klartext-YAML-Dokumenten erstellt. Weitere Informationen zur Dokumentsyntax finden Sie unter Verwenden von Komponentendokumenten in AWSTOE.

Im Folgenden finden Sie Beispiele für Hello World-Komponentendokumente, mit denen Sie Ihre Dokumente lokal entwickeln können.

`hello-world-windows.yml`.

```
name: Hello World
description: This is Hello World testing document for Windows.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
```

```
        inputs:
          commands:
            - Write-Host 'Hello World from the test phase.'
```

`hello-world-linux.yml`.

```
name: Hello World
description: This is hello world testing document for Linux.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the test phase.'
```

Schritt 4: Validieren von AWSTOE Komponenten

Sie können die Syntax von AWSTOE Komponenten lokal mit der AWSTOE Anwendung überprüfen. Die folgenden Beispiele zeigen den AWSTOE `validate`Anwendungsbefehl , um die Syntax einer Komponente zu überprüfen, ohne sie auszuführen.

> **Note**
>
> Die AWSTOE Anwendung kann nur die Komponentensyntax für das aktuelle Betriebssystem validieren. Wenn Sie beispielsweise `awstoe.exe` unter Windows ausführen, können Sie

die Syntax für ein Linux-Dokument, das das `ExecuteBash` Aktionsmodul verwendet, nicht überprüfen.

Windows

```
C:\> awstoe.exe validate --documents C:\Users\user\Documents\hello-world.yml
```

Linux

```
$ awstoe validate --documents /home/user/hello-world.yml
```

Schritt 5: Ausführen von AWSTOE Komponenten

Die AWSTOE Anwendung kann eine oder mehrere Phasen bestimmter Dokumente mit dem `--phases` Befehlszeilenargument ausführen. Unterstützte Werte für `--phases` sind `build`, `validate`, und `test`. Mehrere Phasenwerte können als kommagetrennte Werte eingegeben werden.

Wenn Sie eine Liste von Phasen angeben, führt die AWSTOE Anwendung nacheinander die angegebenen Phasen jedes Dokuments aus. führt beispielsweise AWSTOE die `validate` Phasen `build` und von aus`document1.yaml`, gefolgt von den `validate` Phasen `build` und von `document2.yaml`.

Um sicherzustellen, dass Ihre Protokolle sicher gespeichert und zur Fehlerbehebung aufbewahrt werden, empfehlen wir, den Protokollspeicher in Amazon S3 zu konfigurieren. In Image Builder wird der Amazon S3-Speicherort für die Veröffentlichung von Protokollen in der Infrastrukturkonfiguration angegeben. Weitere Informationen zur Infrastrukturkonfiguration finden Sie unter Verwalten der Infrastrukturkonfiguration von EC2 Image Builder

Wenn keine Liste von Phasen bereitgestellt wird, führt die AWSTOE Anwendung alle Phasen in der im YAML-Dokument aufgeführten Reihenfolge aus.

Verwenden Sie die folgenden Befehle, um bestimmte Phasen in einem oder mehreren Dokumenten auszuführen.

Einzelphase

```
awstoe run --documents hello-world.yml --phases build
```

Mehrere Phasen

```
awstoe run --documents hello-world.yml --phases build,test
```

Dokumentausführung

Alle Phasen in einem einzigen Dokument ausführen

```
awstoe run --documents documentName.yaml
```

Alle Phasen in mehreren Dokumenten ausführen

```
awstoe run --documents documentName1.yaml,documentName2.yaml
```

Geben Sie Amazon S3-Informationen ein, um AWSTOE Protokolle aus einem benutzerdefinierten lokalen Pfad hochzuladen (empfohlen)

```
awstoe run --documents documentName.yaml --log-s3-bucket-name <S3Bucket> --log-s3-key-
prefix <S3KeyPrefix> --log-s3-bucket-owner <S3BucketOwner> --log-directory <local_path>
```

Führen Sie alle Phasen in einem einzigen Dokument aus und zeigen Sie alle Protokolle in der Konsole an

```
awstoe run --documents documentName.yaml --trace
```

-Beispielbefehl

```
awstoe run --documents s3://bucket/key/doc.yaml --phases build,validate
```

Ausführen eines Dokuments mit eindeutiger ID

```
awstoe run --documents <documentName>.yaml --execution-id <user provided id> --phases
 <comma separated list of phases>
```

Erhalten Sie Hilfe mit AWSTOE

```
awstoe --help
```

Verwenden von Komponentendokumenten in AWSTOE

Um eine Komponente mit AWS Task Orchestrator and Executor (AWSTOE) zu erstellen, müssen Sie ein YAML-basiertes Dokument bereitstellen, das die Phasen und Schritte darstellt, die für die von Ihnen erstellte Komponente gelten. AWS-Services verwenden Sie Ihre Komponente, wenn sie ein neues Amazon Machine Image (AMI) oder Container-Image erstellen.

Themen

Workflow für Komponentendokumente

Das AWSTOE Komponentendokument verwendet Phasen und Schritte, um verwandte Aufgaben zu gruppieren und diese Aufgaben in einem logischen Workflow für die Komponente zu organisieren.

> **Tip**
>
> Der Service, der Ihre Komponente zum Erstellen eines Images verwendet, implementiert möglicherweise Regeln darüber, welche Phasen für seinen Build-Prozess verwendet werden sollen und wann diese Phasen ausgeführt werden dürfen. Dies ist wichtig, wenn Sie Ihre Komponente entwerfen.

Phasen

Phasen stellen den Fortschritt Ihres Workflows während des Image-Erstellungsprozesses dar. Beispielsweise verwendet der Image-Builder-Service während seiner Build-Phase die `validate` Phasen `build` und für die von ihm erstellten Images. Es verwendet die `container-host-test` Phasen `test` und während seiner Testphase, um sicherzustellen, dass der Image-Snapshot oder

das Container-Image die erwarteten Ergebnisse generiert, bevor das endgültige AMI erstellt oder das Container-Image verteilt wird.

Wenn die Komponente ausgeführt wird, werden die zugehörigen Befehle für jede Phase in der Reihenfolge angewendet, in der sie im Komponentendokument angezeigt werden.

Regeln für Phasen

- Jeder Phasenname muss innerhalb eines Dokuments eindeutig sein.
- Sie können viele Phasen in Ihrem Dokument definieren.
- Sie müssen mindestens eine der folgenden Phasen in Ihr Dokument aufnehmen:
  - build – für Image Builder wird diese Phase im Allgemeinen während der Build-Phase verwendet.
  - validate – Für Image Builder wird diese Phase im Allgemeinen während der Build-Phase verwendet.
  - test – Für Image Builder wird diese Phase im Allgemeinen während der Testphase verwendet.
- Phasen werden immer in der Reihenfolge ausgeführt, in der sie im Dokument definiert sind. Die Reihenfolge, in der sie für AWSTOE Befehle in der angegeben werden, AWS CLI hat keine Auswirkungen.

Schritte

Schritte sind einzelne Arbeitseinheiten, die den Workflow innerhalb jeder Phase definieren. Die Schritte werden nacheinander ausgeführt. Die Eingabe oder Ausgabe für einen Schritt kann jedoch auch in einen nachfolgenden Schritt als Eingabe einfließen. Dies wird als „Verkettung“ bezeichnet.

Regeln für Schritte

- Der Schrittname muss für die -Phase eindeutig sein.
- Der Schritt muss eine unterstützte Aktion (Aktionsmodul) verwenden, die einen Beendigungscode zurückgibt.

  Eine vollständige Liste der unterstützten Aktionsmodule, ihrer Funktionsweise, Eingabe-/Ausgabewerte und Beispiele finden Sie unter Vom AWSTOE Komponentenmanager unterstützte Aktionsmodule.

Komponentenprotokollierung

AWSTOE erstellt bei jeder Ausführung Ihrer Komponente einen neuen Protokollordner auf den EC2-Instances, die zum Erstellen und Testen eines neuen Images verwendet werden. Bei Container-Images wird der Protokollordner im Container gespeichert.

Zur Unterstützung bei der Fehlerbehebung, wenn während der Image-Erstellung etwas schief geht, werden das Eingabedokument und alle Ausgabedateien, die während der Ausführung der Komponente AWSTOE erstellt, im Protokollordner gespeichert.

Der Name des Protokollordners besteht aus den folgenden Teilen:

1. Protokollverzeichnis – Wenn ein Service eine AWSTOE Komponente ausführt, übergibt er das Protokollverzeichnis zusammen mit anderen Einstellungen für den Befehl. In den folgenden Beispielen zeigen wir das Protokolldateiformat, das Image Builder verwendet.
   - Linux: `/var/lib/amazon/toe/`
   - Windows: `$env:ProgramFiles\Amazon\TaskOrchestratorAndExecutor\`
2. Dateipräfix – Dies ist ein Standardpräfix, das für alle Komponenten verwendet wird: „`TOE_`“.
3. Laufzeit – Dies ist ein Zeitstempel im Format JJJJ-MM-TT_HH-MM-SS_UTC-0.
4. Ausführungs-ID – Dies ist die GUID, die zugewiesen wird, wenn eine oder mehrere Komponenten AWSTOE ausführt.

Beispiel: *`/var/lib/amazon/toe/`*`TOE_`*`2021-07-01_12-34-56_UTC-0_a1bcd2e3-45f6-789a-bcde-0fa1b2c3def4`*

AWSTOE speichert die folgenden Core-Dateien im Protokollordner:

Eingabedateien

- document.yaml – Das Dokument, das als Eingabe für den Befehl verwendet wird. Nachdem die Komponente ausgeführt wurde, wird diese Datei als Artefakt gespeichert.

Ausgabedateien

- application.log – Das Anwendungsprotokoll enthält Informationen auf Debug-Ebene mit Zeitstempeln von AWSTOE darüber, was während der Ausführung der Komponente passiert.

- detailedoutput.json – Diese JSON-Datei enthält detaillierte Informationen über den Ausführungsstatus, Eingaben, Ausgaben und Fehler für alle Dokumente, Phasen und Schritte, die für die Komponente gelten, während sie ausgeführt wird.
- console.log – Das Konsolenprotokoll enthält alle Informationen zum Standardausgang (stdout) und zum Standardfehler (stderr), die während der Ausführung der Komponente in die Konsole AWSTOEschreibt.
- chaining.json – Diese JSON-Datei stellt Optimierungen dar, die zur Auflösung verketteter Ausdrücke AWSTOE angewendet wurden.

> **Note**
>
> Der Protokollordner kann auch andere temporäre Dateien enthalten, die hier nicht behandelt werden.

Eingabe- und Ausgabeverkettung

Die AWSTOE Konfigurationsverwaltungsanwendung bietet eine Funktion zum Verketten von Eingaben und Ausgaben, indem Referenzen in den folgenden Formaten geschrieben werden:

```
{{ phase_name.step_name.inputs/outputs.variable }}
```

or

```
{{ phase_name.step_name.inputs/outputs[index].variable }}
```

Mit der Verkettungsfunktion können Sie Code recyceln und die Wartungsbarkeit des Dokuments verbessern.

Regeln für die Verkettung

- Verkettungsausdrücke können nur im Eingabebereich jedes Schritts verwendet werden.
- Anweisungen mit verketteten Ausdrücken müssen in Anführungszeichen eingeschlossen werden. Beispielsweise:
  - Ungültiger Ausdruck: `echo {{ phase.step.inputs.variable }}`
  - Gültiger Ausdruck: `"echo {{ phase.step.inputs.variable }}"`
  - Gültiger Ausdruck: `'echo {{ phase.step.inputs.variable }}'`

- Verkettungsausdrücke können auf Variablen aus anderen Schritten und Phasen im selben Dokument verweisen. Der aufrufende Service kann jedoch Regeln enthalten, die erfordern, dass Verkettungsausdrücke nur im Kontext einer einzelnen Phase ausgeführt werden. Image Builder unterstützt beispielsweise keine Verkettung von der Build-Phase zur Testphase , da jede Phase unabhängig ausgeführt wird.
- Indizes in verketteten Ausdrücken folgen der nullbasierten Indizierung. Der Index beginnt mit Null (0), um auf das erste Element zu verweisen.

Beispiele

Um im zweiten Eintrag des folgenden Beispielschritts auf die Quellvariable zu verweisen, lautet das Verkettungsmuster `{{ build.SampleS3Download.inputs[1].source }}`.

```
phases:
  -
    name: 'build'
    steps:
      -
        name: SampleS3Download
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          -
            source: 's3://sample-bucket/sample1.ps1'
            destination: 'C:\sample1.ps1'
          -
            source: 's3://sample-bucket/sample2.ps1'
            destination: 'C:\sample2.ps1'
```

Um auf die Ausgabevariable (gleich „Hello“) des folgenden Beispielschritts zu verweisen, ist das Verkettungsmuster `{{ build.SamplePowerShellStep.outputs.stdout }}`.

```
phases:
  -
    name: 'build'
    steps:
      -
        name: SamplePowerShellStep
        action: ExecutePowerShell
```

```
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          commands:
            - 'Write-Host "Hello"'
```

Dokumentschema und Definitionen

Im Folgenden finden Sie das YAML-Schema für ein Dokument.

```
name: (optional)
description: (optional)
schemaVersion: "string"

phases:
  - name: "string"
    steps:
        - name: "string"
          action: "string"
          timeoutSeconds: integer
          onFailure: "Abort|Continue|Ignore"
          maxAttempts: integer
          inputs:
```

Die Schemadefinitionen für ein Dokument lauten wie folgt.

Feld	Beschreibung	Typ	Erforderlich
Name	Name des Dokuments.	String	Nein
description	Beschreibung des Dokuments.	String	Nein
schemaVersion	Schemaversion des Dokuments, derzeit 1.0.	String	Ja
phases	Eine Liste von Phasen mit ihren Schritten.	Auflisten	Ja

Die Schemadefinitionen für eine Phase lauten wie folgt.

Feld	Beschreibung	Typ	Erforderlich
Name	Name der -Phase.	String	Ja
steps	Liste der Schritte in der -Phase.	Auflisten	Ja

Die Schemadefinitionen für einen Schritt lauten wie folgt.

Feld	Beschreibung	Typ	Erforderlich	Standardwert
Name	Benutzerd efinierter Name für den Schritt.	String		
action	Schlüsselwort, das sich auf das Modul bezieht, das den Schritt ausführt.	String		
timeoutSeconds	Anzahl der Sekunden, die der Schritt ausgeführt wird, bevor er fehlschlägt oder wiederholt wird. Außerdem unterstützt den Wert -1, was auf ein unendlich es Timeout hinweist. 0 und	Ganzzahl	Nein	7 200 Sekunden (120 Minuten)

Feld	Beschreibung	Typ	Erforderlich	Standardwert
	andere negative Werte sind nicht zulässig.			

Feld	Beschreibung	Typ	Erforderlich	Standardwert
onFailure	Gibt an, was der Schritt im Falle eines Ausfalls tun soll. Gültige Werte sind: • Abbruch – Der Schritt schlägt nach der maximalen Anzahl von Versuchen fehl und wird nicht mehr ausgeführt. Legt den Status für Phase und Dokument auf fest`Failed`. • Weiter – Schlägt den Schritt nach der maximalen Anzahl von Versuchen fehl und führt weiterhin die verbleibenden Schritte aus. Legt den Status für Phase und Dokument auf fest`Failed`.	String	Nein	Abbrechen

Feld	Beschreibung	Typ	Erforderlich	Standardwert
	• Ignoriere n – Legt den Schritt `IgnoredFailure` nach der maximalen Anzahl fehlgeschlagener Versuche auf fest und führt weiterhin die verbleibenden Schritte aus. Legt den Status für Phase und Dokument auf fest`SuccessWithIgnoredFailure`.			
maxAttempts	Maximale Anzahl zulässiger Versuche, bevor der Schritt fehlschlägt.	Ganzzahl	Nein	1
inputs	Enthält Parameter, die das Aktionsmodul benötigt, um den Schritt auszuführen.	Diktat	Ja	

Schemata für Dokumentbeispiele

Im Folgenden finden Sie ein Beispiel für ein Dokumentschema zum Installieren aller verfügbaren Windows-Updates, zum Ausführen eines Konfigurationsskripts, zum Validieren der Änderungen vor der Erstellung des AMI und zum Testen der Änderungen nach der Erstellung des AMI.

```
name: RunConfig_UpdateWindows
description: 'This document will install all available Windows updates and run a config
 script. It will then validate the changes before an AMI is created. Then after AMI
 creation, it will test all the changes.'
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: DownloadConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/config.ps1'
            destination: 'C:\config.ps1'

      - name: RunConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          file: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: Cleanup
        action: DeleteFile
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - path: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: RebootAfterConfigApplied
        action: Reboot
        inputs:
          delaySeconds: 60
```

```
      - name: InstallWindowsUpdates
        action: UpdateOS

  - name: validate
    steps:
      - name: DownloadTestConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/testConfig.ps1'
            destination: 'C:\testConfig.ps1'

      - name: ValidateConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          file: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

      - name: Cleanup
        action: DeleteFile
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - path: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

  - name: test
    steps:
      - name: DownloadTestConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/testConfig.ps1'
            destination: 'C:\testConfig.ps1'

      - name: ValidateConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
```

```
        maxAttempts: 3
        inputs:
          file: '{{test.DownloadTestConfigScript.inputs[0].destination}}'
```

Im Folgenden finden Sie ein Beispiel für ein Dokumentschema zum Herunterladen und Ausführen einer benutzerdefinierten Linux-Binärdatei.

```
name: LinuxBin
description: Download and run a custom Linux binary file.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://<replaceable>mybucket</replaceable>/
<replaceable>myapplication</replaceable>
            destination: /tmp/<replaceable>myapplication</replaceable>
      - name: Enable
        action: ExecuteBash
        onFailure: Continue
        inputs:
          commands:
            - 'chmod u+x {{ build.Download.inputs[0].destination }}'
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '--install'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Im Folgenden finden Sie ein Beispiel für ein Dokumentschema zur Installation der AWS CLI auf einer Windows-Instance mithilfe der Setup-Datei.

```
name: InstallCLISetUp
description: Install &CLI; using the setup file
```

```
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLISetup.exe
            destination: C:\Windows\temp\AWSCLISetup.exe
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '/install'
            - '/quiet'
            - '/norestart'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Im Folgenden finden Sie ein Beispiel für ein Dokumentschema zur Installation des AWS CLI mit dem MSI-Installationsprogramm.

```
name: InstallCLIMSI
description: Install &CLI; using the MSI installer
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLI64PY3.msi
            destination: C:\Windows\temp\AWSCLI64PY3.msi
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: 'C:\Windows\System32\msiexec.exe'
          arguments:
```

```
            - '/i'
            - '{{ build.Download.inputs[0].destination }}'
            - '/quiet'
            - '/norestart'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Definieren und Referenzieren von Variablen in AWSTOE

Variablen bieten eine Möglichkeit, Daten mit aussagekräftigen Namen zu kennzeichnen, die in einer Anwendung verwendet werden können. Sie können benutzerdefinierte Variablen mit einfachen und lesbaren Formaten für komplexe Workflows definieren und sie im Dokument mit der YAML-Anwendungskomponente für eine -AWSTOEKomponente referenzieren.

Dieser Abschnitt enthält Informationen, die Ihnen helfen, Variablen für Ihre AWSTOE Komponente im YAML-Anwendungskomponentendokument zu definieren, einschließlich Syntax, Namenseinschränkungen und Beispielen.

Parameter

Parameter sind veränderbare Variablen mit Einstellungen, die die aufrufende Anwendung zur Laufzeit bereitstellen kann. Sie können Parameter im -`parameters`Abschnitt des YAML-Dokuments definieren.

Regeln für Parameternamen

- Der Name muss zwischen 3 und 128 Zeichen lang sein.
- Der Name darf nur alphanumerische Zeichen (a-z, A-Z, 0-9), Bindestriche (-) oder Unterstriche (_) enthalten.
- Der Name muss innerhalb des Dokuments eindeutig sein.
- Der Name muss als YAML-Zeichenfolge angegeben werden.

Syntax

```
parameters:
  - <name>:
      type: <parameter type>
      default: <parameter value>
```

```
      description: <parameter description>
```

Tastenname	Erforderlich	Beschreibung
`name`	Ja	Der Name des Parameters. Muss für das Dokument eindeutig sein (es darf nicht mit anderen Parameternamen oder Konstanten identisch sein).
`type`	Ja	Der Datentyp des Parameters. Zu den unterstützten Typen gehören: `string`.
`default`	Nein	Der Standardwert für den Parameter.
`description`	Nein	Beschreibt den -Parameter.

Referenzparameterwerte in einem Dokument

Sie können Parameter in Schritt- oder Schleifeneingaben innerhalb Ihres YAML-Dokuments wie folgt referenzieren:

- Bei Parameterreferenzen wird zwischen Groß- und Kleinschreibung unterschieden, und der Name muss genau übereinstimmen.
- Der Name muss in doppelte geschweifte Klammern eingeschlossen werden`{{MyParameter}}`.
- Leerzeichen sind innerhalb der geschweiften Klammern zulässig und werden automatisch gekürzt. Beispielsweise sind alle der folgenden Referenzen gültig:

  `{{ MyParameter }}`, `{{ MyParameter}}`, `{{MyParameter }}`, `{{MyParameter}}`
- Die Referenz im YAML-Dokument muss als Zeichenfolge angegeben werden (in einfache oder doppelte Anführungszeichen eingeschlossen).

  Zum Beispiel: - `{{ MyParameter }}` ist nicht gültig, da es nicht als Zeichenfolge identifiziert wird.

Die folgenden Referenzen sind jedoch beide gültig: - '{{ *MyParameter* }}' und - "{{ *MyParameter* }}".

Beispiele

Die folgenden Beispiele zeigen, wie Sie Parameter in Ihrem YAML-Dokument verwenden:

- Verweisen Sie auf einen Parameter in Schritteingaben:

```
name: Download AWS CLI version 2
schemaVersion: 1.0
parameters:
  - Source:
      type: string
      default: 'https://awscli.amazonaws.com/AWSCLIV2.msi'
      description: The AWS CLI installer source URL.
phases:
  - name: build
    steps:
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

- Verweisen Sie auf einen Parameter in Schleifeneingaben:

```
name: PingHosts
schemaVersion: 1.0
parameters:
  - Hosts:
      type: string
      default: 127.0.0.1,amazon.com
      description: A comma separated list of hosts to ping.
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
            list: '{{ Hosts }}'
```

```
            delimiter: ','
        inputs:
          commands:
            - ping -c 4 {{ loop.value }}
```

Überschreiben von Parametern zur Laufzeit

Sie können die `--parameters` Option aus dem AWS CLI mit einem Schlüssel-Wert-Paar verwenden, um zur Laufzeit einen Parameterwert festzulegen.

- Geben Sie das Parameter-Schlüssel-Wert-Paar als Namen und Wert an, getrennt durch ein Gleichheitszeichen (<name>=<value>).
- Mehrere Parameter müssen durch ein Komma getrennt werden.
- Parameternamen, die nicht im YAML-Komponentendokument gefunden werden, werden ignoriert.
- Der Parametername und der Wert sind beide erforderlich.

> **Important**
>
> Komponentenparameter sind Klartextwerte und werden in protokolliertAWS CloudTrail. Wir empfehlen Ihnen, AWS Secrets Manager oder den AWS Systems Manager Parameter Store zu verwenden, um Ihre Secrets zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch. Weitere Informationen zu AWS Systems Manager Parameter Store finden Sie unter AWS Systems Manager Parameter Store im AWS Systems Manager -Benutzerhandbuch.

Syntax

```
--parameters name1=value1,name2=value2...
```

CLI-Option	Erforderlich	Beschreibung
Name *der* --Parameter = *Wert* ,...	Nein	Diese Option verwendet eine Liste von Schlüssel-Wert-Paaren mit dem Parameternamen als Schlüssel.

Beispiele

Die folgenden Beispiele zeigen, wie Sie Parameter in Ihrem YAML-Dokument verwenden:

- Das in dieser `--parameter` Option angegebene Parameter-Schlüssel-Wert-Paar ist ungültig:

```
--parameters ntp-server=
```

- Legen Sie ein Parameter-Schlüssel-Wert-Paar mit der `--parameter` Option in der festAWS CLI:

```
--parameters ntp-server=ntp-server-windows-qe.us-east1.amazon.com
```

- Legen Sie mehrere Parameter-Schlüssel-Wert-Paare mit der `--parameter` Option in der festAWS CLI:

```
--parameters ntp-server=ntp-server.amazon.com,http-url=https://internal-us-
east1.amazon.com
```

Konstanten

Konstanten sind unveränderliche Variablen, die nach der Definition nicht mehr geändert oder überschrieben werden können. Konstanten können mithilfe von Werten im -`constants`Abschnitt eines -AWSTOEDokuments definiert werden.

Regeln für konstante Namen

- Der Name muss zwischen 3 und 128 Zeichen lang sein.
- Der Name darf nur alphanumerische Zeichen (a-z, A-Z, 0-9), Bindestriche (-) oder Unterstriche (_) enthalten.
- Der Name muss innerhalb des Dokuments eindeutig sein.
- Der Name muss als YAML-Zeichenfolge angegeben werden.

Syntax

```
constants:
  - <name>:
      type: <constant type>
      value: <constant value>
```

Tastenname	Erforderlich	Beschreibung
`name`	Ja	Name der Konstante. Muss für das Dokument eindeutig sein (es darf nicht mit anderen Parameternamen oder Konstanten identisch sein).
`value`	Ja	Wert der Konstante.
`type`	Ja	Typ der Konstante. Der unterstützte Typ ist `string`.

Referenzieren konstanter Werte in einem Dokument

Sie können in Schritt- oder Schleifeneingaben innerhalb Ihres YAML-Dokuments wie folgt auf Konstanten verweisen:

- Bei konstanten Verweisen wird zwischen Groß- und Kleinschreibung unterschieden, und der Name muss genau übereinstimmen.
- Der Name muss in doppelte geschweifte Klammern eingeschlossen werden`{{`*`MyConstant`*`}}`.
- Leerzeichen sind innerhalb der geschweiften Klammern zulässig und werden automatisch gekürzt. Beispielsweise sind alle der folgenden Referenzen gültig:

  `{{` *`MyConstant`* `}}`, `{{` *`MyConstant`*`}}`, `{{`*`MyConstant`* `}}`, `{{`*`MyConstant`*`}}`
- Die Referenz im YAML-Dokument muss als Zeichenfolge angegeben werden (in einfache oder doppelte Anführungszeichen eingeschlossen).

  Zum Beispiel: `- {{` *`MyConstant`* `}}` ist nicht gültig, da es nicht als Zeichenfolge identifiziert wird.

  Die folgenden Referenzen sind jedoch beide gültig: `- '{{` *`MyConstant`* `}}'` und `- "{{` *`MyConstant`* `}}"`.

Beispiele

Konstant referenziert in Schritteingaben

```
name: Download AWS CLI version 2
```

```
schemaVersion: 1.0
constants:
  - Source:
      type: string
      value: https://awscli.amazonaws.com/AWSCLIV2.msi
phases:
  - name: build
    steps:
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

Konstant referenziert in Schleifeneingaben

```
name: PingHosts
schemaVersion: 1.0
constants:
  - Hosts:
      type: string
      value: 127.0.0.1,amazon.com
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
            list: '{{ Hosts }}'
            delimiter: ','
        inputs:
          commands:
            - ping -c 4 {{ loop.value }}
```

Verwenden von Schleifenkonstrukten in AWSTOE

Dieser Abschnitt enthält Informationen, die Ihnen beim Erstellen von Schleifenkonstrukten in der helfenAWSTOE. Schleifenkonstrukte definieren eine wiederholte Reihenfolge von Anweisungen. Sie können die folgenden Arten von Looping-Konstrukten in verwendenAWSTOE:

- `for` -Konstrukte – Iterieren Sie über eine begrenzte Folge von ganzen Zahlen.

- `forEach` Konstrukte
  - `forEach` -Schleife mit Eingabeliste – Iteriert über eine endliche Sammlung von Zeichenfolgen.
  - `forEach` -Schleife mit getrennter Liste – Iteriert über eine endliche Sammlung von Zeichenfolgen, die durch ein Trennzeichen verbunden sind.

> **Note**
>
> Schleifenkonstrukte unterstützen nur Zeichenfolgen-Datentypen.

Themen zum Schleifen von Konstrukten

- Referenz-Iterationsvariablen
- Arten von Looping-Konstrukten
- Schrittfelder
- Schritt- und Iterationsausgaben

Referenz-Iterationsvariablen

Um auf den Index und den Wert der aktuellen Iterationsvariablen zu verweisen, `{{ loop.* }}` muss der Referenzausdruck innerhalb des Eingabetexts eines Schritts verwendet werden, der ein Schleifenkonstrukt enthält. Dieser Ausdruck kann nicht verwendet werden, um auf die Iterationsvariablen des Schleifen-Konstrukts eines anderen Schritts zu verweisen.

Der Referenzausdruck besteht aus den folgenden Elementen:

- `{{ loop.index }}` – Die Ordinalposition der aktuellen Iteration, die bei indiziert wird`0`.
- `{{ loop.value }}` – Der Wert, der der aktuellen Iterationsvariablen zugeordnet ist.

Schleifennamen

Alle Schleifenkonstrukte haben ein optionales Namensfeld zur Identifizierung. Wenn ein Loop-Name angegeben wird, kann er verwendet werden, um auf Iterationsvariablen im Eingabetext des Schritts zu verweisen. Um auf die Iterationsindizes und Werte einer benannten Schleife zu verweisen, verwenden Sie `{{ <loop_name>.* }}` mit `{{ loop.* }}` im Eingabetext des Schritts. Dieser Ausdruck kann nicht verwendet werden, um auf das benannte Looping-Konstrukt eines anderen Schritts zu verweisen.

Der Referenzausdruck besteht aus den folgenden Elementen:

- `{{ <loop_name>.index }}` – Die Ordinalposition der aktuellen Iteration der benannten Schleife, die bei indiziert wird`0`.
- `{{ <loop_name>.value }}` – Der Wert, der der aktuellen Iterationsvariable der benannten Schleife zugeordnet ist.

Referenzausdrücke auflösen

Die AWSTOE löst Referenzausdrücke wie folgt auf:

- `{{ <loop_name>.* }}` – AWSTOE löst diesen Ausdruck mit der folgenden Logik auf:
  - Wenn die Schleife des aktuell ausgeführten Schritts mit dem `<loop_name>` Wert übereinstimmt, wird der Referenzausdruck in das Schleifenkonstrukt des aktuell ausgeführten Schritts aufgelöst.
  - `<loop_name>` wird in das benannte Looping-Konstrukt aufgelöst, wenn es im aktuell ausgeführten Schritt erscheint.
- `{{ loop.* }}` – AWSTOE löst den Ausdruck mithilfe des Schleifen-Konstrukts auf, das im aktuell ausgeführten Schritt definiert ist.

Wenn Referenzausdrücke innerhalb eines Schritts verwendet werden, der keine Schleife enthält, löst die Ausdrücke AWSTOE nicht auf und sie werden im Schritt ohne Ersatz angezeigt.

> **Note**
>
> Referenzausdrücke müssen in doppelte Anführungszeichen eingeschlossen werden, damit sie vom YAML-Compiler korrekt interpretiert werden können.

Arten von Looping-Konstrukten

Dieser Abschnitt enthält Informationen und Beispiele zum Schleifen von Konstrukttypen, die in der verwendet werden könnenAWSTOE.

Schleifen-Konstrukttypen

- for Schleife
- forEach -Schleife mit Eingabeliste

- forEach -Schleife mit getrennter Liste

`for` Schleife

Die `for` Schleife iteriert auf einem Bereich von Ganzzahlen, die innerhalb einer durch den Anfang und das Ende der Variablen definierten Grenze angegeben sind. Die iterierenden Werte befinden sich im Satz `[start, end]` und enthalten Grenzwerte.

AWSTOE überprüft die `updateBy` Werte `start`, und `end`, um sicherzustellen, dass die Kombination nicht zu einer Endlosschleife führt.

`for` -Schleifenschema

```
name: "StepName"
action: "ActionModule"
loop:
  name: "string"
  for:
    start: int
    end: int
    updateBy: int
inputs:
  ...
```

`for` Loop-Eingabe

Feld	Beschreibung	Typ	Erforderlich	Standard
`name`	Eindeutiger Name der Schleife. Er muss im Vergleich zu anderen Loop-Namen in derselben Phase eindeutig sein.	String	Nein	""
`start`	Startwert der Iteration.	Ganzzahl	Ja	–

Feld	Beschreibung	Typ	Erforderlich	Standard
	Akzeptiert keine verkettenden Ausdrücke.			
`end`	Endwert der Iteration. Akzeptiert keine verkettenden Ausdrücke.	Ganzzahl	Ja	–
`updateBy`	Unterschied, um den ein iterierender Wert durch Addition aktualisiert wird. Es muss ein negativer oder positiver Wert ungleich Null sein. Akzeptiert keine verketten den Ausdrücke.	Ganzzahl	Ja	–

`for` Beispiel für eine Loop-Eingabe

```
name: "CalculateFileUploadLatencies"
action: "ExecutePowerShell"
loop:
  for:
    start: 100000
    end: 1000000
    updateBy: 100000
inputs:
  commands:
    - |
      $f = new-object System.IO.FileStream c:\temp\test{{ loop.index }}.txt, Create,
 ReadWrite
```

```
      $f.SetLength({{ loop.value }}MB)
      $f.Close()
    - c:\users\administrator\downloads\latencyTest.exe --file c:\temp
\test{{ loop.index }}.txt
    - AWS s3 cp c:\users\administrator\downloads\latencyMetrics.json s3://bucket/
latencyMetrics.json
    - |
      Remove-Item -Path c:\temp\test{{ loop.index }}.txt
      Remove-Item -Path c:\users\administrator\downloads\latencyMetrics.json
```

forEach -Schleife mit Eingabeliste

Die `forEach` Schleife iteriert anhand einer expliziten Liste von Werten, bei denen es sich um Zeichenfolgen und verkettete Ausdrücke handeln kann.

`forEach` -Schleife mit Eingabelistenschema

```
name: "StepName"
action: "ActionModule"
loop:
  name: "string"
  forEach:
    - "string"
inputs:
  ...
```

forEach -Schleife mit Eingabelisteneingabe

Feld	Beschreibung	Typ	Erforderlich	Standard
`name`	Eindeutiger Name der Schleife. Er muss im Vergleich zu anderen Loop-Namen in derselben Phase eindeutig sein.	String	Nein	""

Feld	Beschreibung	Typ	Erforderlich	Standard
Liste der `forEach` Loop-Zeichenfolgen	Liste der Zeichenfolgen für die Iteration . Akzeptier t verkettete Ausdrücke als Zeichenfo lgen in der Liste. Verkettet e Ausdrücke müssen in doppelte Anführung szeichen gesetzt werden, damit der YAML-Comp iler sie korrekt interpretieren kann.	Liste von Zeichenfolgen	Ja	–

`forEach` -Schleife mit Eingabeliste, Beispiel 1

```
name: "ExecuteCustomScripts"
action: "ExecuteBash"
loop:
  name: BatchExecLoop
  forEach:
    - /tmp/script1.sh
    - /tmp/script2.sh
    - /tmp/script3.sh
inputs:
  commands:
    - echo "Count {{ BatchExecLoop.index }}"
    - sh "{{ loop.value }}"
    - |
      retVal=$?
```

```
      if [ $retVal -ne 0 ]; then
        echo "Failed"
      else
        echo "Passed"
      fi
```

`forEach` -Schleife mit Eingabeliste, Beispiel 2

```
name: "RunMSIWithDifferentArgs"
action: "ExecuteBinary"
loop:
  name: MultiArgLoop
  forEach:
    - "ARG1=C:\Users ARG2=1"
    - "ARG1=C:\Users"
    - "ARG1=C:\Users ARG3=C:\Users\Administrator\Documents\f1.txt"
inputs:
  commands:
    path: "c:\users\administrator\downloads\runner.exe"
    args:
      - "{{ MultiArgLoop.value }}"
```

`forEach` -Schleife mit Eingabeliste, Beispiel 3

```
name: "DownloadAllBinaries"
action: "S3Download"
loop:
  name: MultiArgLoop
  forEach:
    - "bin1.exe"
    - "bin10.exe"
    - "bin5.exe"
inputs:
  -
    source: "s3://bucket/{{ loop.value }}"
    destination: "c:\temp\{{ loop.value }}"
```

`forEach` -Schleife mit getrennter Liste

Die Schleife iteriert über eine Zeichenfolge, die durch ein Trennzeichen getrennte Werte enthält. Um über die Bestandteile der Zeichenfolge zu iterieren, AWSTOE verwendet das Trennzeichen, um die Zeichenfolge in ein Array aufzuteilen, das für die Iteration geeignet ist.

`forEach` -Schleife mit getrenntem Listenschema

```
name: "StepName"
action: "ActionModule"
loop:
  name: "string"
  forEach:
    list: "string"
    delimiter: ".,;:\n\t -_"
inputs:
  ...
```

`forEach` -Schleife mit getrennter Listeneingabe

Feld	Beschreibung	Typ	Erforderlich	Standard
`name`	Eindeutiger Name, der der Schleife gegeben wird. Er sollte im Vergleich zu anderen Loop-Namen in derselben Phase eindeutig sein.	String	Nein	""
`list`	Eine Zeichenfolge, die aus zusammenhängenden Zeichenfolgen besteht, die durch ein gemeinsames Trennzeichen verbunden sind. Akzeptiert auch verkettet	String	Ja	–

Feld	Beschreibung	Typ	Erforderlich	Standard
	e Ausdrücke . Stellen Sie bei verketteten Ausdrücken sicher, dass diese zur korrekten Interpretation durch den YAML-Compiler in doppelte Anführungszeichen eingeschlossen sind.			

Feld	Beschreibung	Typ	Erforderlich	Standard
`delimiter`	Zeichen, das zum Trennen von Zeichenfolgen innerhalb eines Blocks verwendet wird. Standard ist das Kommazeichen. Aus der angegebenen Liste ist nur ein Trennzeichen zulässig: • Punkt: `"."` • Komma: `","` • Semikolon: `";"` • Kolon: `":"` • Neue Zeile: `"\n"` • Registerkarte: `"\t"` • Bereich: `" "` • Bindestrich: `"-"` • Unterstrich: `"_"` Verkettungsausdrücke können nicht	String	Nein	Komma: `","`

Feld	Beschreibung	Typ	Erforderlich	Standard
	verwendet werden.			

> **Note**
>
> Der Wert von `list` wird als unveränderliche Zeichenfolge behandelt. Wenn die Quelle von während der Laufzeit geändert `list` wird, wird sie während der Ausführung nicht wiedergegeben.

`forEach` -Schleife mit getrennter Liste, Beispiel 1

```
// Uses chaning expression ({{ <phase_name>.<step_name>.inputs/outputs.<var_name> }})
// to refer to another step's input/output variables for code re-use.
name: "RunMSIs"
action: "ExecuteBinary"
loop:
  forEach:
    list: "{{ build.GetAllMSIPathsForInstallation.outputs.stdout }}"
    delimiter: "\n"
inputs:
  commands:
    path: "{{ loop.value }}"
```

`forEach` -Schleife mit getrennter Liste, Beispiel 2

```
name: "UploadMetricFiles"
action: "S3Upload"
loop:
  forEach:
    list: "/tmp/m1.txt,/tmp/m2.txt,/tmp/m3.txt,..."
inputs:
  commands:
    -
      source: "{{ loop.value }}"
      destination: "s3://bucket/key/{{ loop.value }}"
```

Schrittfelder

Schleifen sind Teil eines Schritts. Jedes Feld im Zusammenhang mit der Ausführung eines Schritts wird nicht auf einzelne Iterationen angewendet. Schrittfelder gelten nur auf Schrittebene, wie folgt:

- timeoutSeconds – Alle Iterationen der Schleife müssen innerhalb des in diesem Feld angegebenen Zeitraums ausgeführt werden. Wenn die Schleifenausführung abläuft, AWSTOE führt die Wiederholungsrichtlinie des Schritts aus und setzt den Timeout-Parameter für jeden neuen Versuch zurück. Wenn die Schleifenausführung den Timeout-Wert überschreitet, nachdem die maximale Anzahl von Wiederholungen erreicht wurde, gibt die Fehlermeldung des Schritts an, dass die Schleifenausführung eine Zeitüberschreitung hatte.
- onFailure – Die Fehlerbehandlung wird wie folgt auf den Schritt angewendet:
  - Wenn onFailure auf gesetzt ist`Abort`, AWSTOE beendet die Schleife und wiederholt den Schritt gemäß der Wiederholungsrichtlinie. Nach der maximalen Anzahl von Wiederholungsversuchen AWSTOE markiert den aktuellen Schritt als fehlgeschlagen und stoppt die Ausführung des Prozesses.

    AWSTOE legt den Statuscode für die übergeordnete Phase und das Dokument auf fest`Failed`.

    > **Note**
    >
    > Nach dem fehlgeschlagenen Schritt werden keine weiteren Schritte ausgeführt.

  - Wenn onFailure auf gesetzt ist`Continue`, AWSTOE beendet die Schleife und wiederholt den Schritt gemäß der Wiederholungsrichtlinie. Nach der maximalen Anzahl von Wiederholungsversuchen AWSTOE markiert den aktuellen Schritt als fehlgeschlagen und fährt mit der Ausführung des nächsten Schritts fort.

    AWSTOE legt den Statuscode für die übergeordnete Phase und das Dokument auf fest`Failed`.
  - Wenn onFailure auf gesetzt ist`Ignore`, AWSTOE beendet die Schleife und wiederholt den Schritt gemäß der Wiederholungsrichtlinie. Nach der maximalen Anzahl von Wiederholungsversuchen AWSTOE markiert den aktuellen Schritt als `IgnoredFailure`und fährt mit der Ausführung des nächsten Schritts fort.

    AWSTOE legt den Statuscode für die übergeordnete Phase und das Dokument auf fest`SuccessWithIgnoredFailure`.

> **Note**
>
> Dies gilt weiterhin als erfolgreicher Lauf, enthält jedoch Informationen, die Sie darüber informieren, dass ein oder mehrere Schritte fehlgeschlagen sind und ignoriert wurden.

- maxAttempts – Bei jedem Wiederholungsversuch werden der gesamte Schritt und alle Iterationen von Anfang an ausgeführt.
- Status – Der Gesamtstatus der Ausführung eines Schritts.`status` stellt nicht den Status einzelner Iterationen dar. Der Status eines Schritts mit Schleifen wird wie folgt bestimmt:
  - Wenn eine einzelne Iteration nicht ausgeführt werden kann, verweist der Status eines Schritts auf einen Fehler.
  - Wenn alle Iterationen erfolgreich sind, verweist der Status eines Schritts auf einen Erfolg.
- startTime – Die Gesamtstartzeit der Ausführung eines Schritts. Stellt nicht die Startzeit einzelner Iterationen dar.
- endTime – Die Gesamtendzeit der Ausführung eines Schritts. Stellt nicht die Endzeit einzelner Iterationen dar.
- failureMessage – Schließt die Iterationsindizes ein, die bei Nicht-Timeout-Fehlern fehlgeschlagen sind. Bei Timeout-Fehlern gibt die Meldung an, dass die Schleifenausführung fehlgeschlagen ist. Einzelne Fehlermeldungen für jede Iteration werden nicht bereitgestellt, um die Größe von Fehlermeldungen zu minimieren.

Schritt- und Iterationsausgaben

Jede Iteration enthält eine Ausgabe. Am Ende einer Schleifenausführung AWSTOE konsolidiert alle erfolgreichen Iterationsausgaben in `detailedOutput.json`. Die konsolidierten Ausgaben sind eine Sortierung von Werten, die zu den entsprechenden Ausgabeschlüsseln gehören, wie im Ausgabeschema des Aktionsmoduls definiert. Das folgende Beispiel zeigt, wie die Ausgaben konsolidiert werden:

Ausgabe von **`ExecuteBash`** für Iteration 1

```
[{"stdout":"Hello"}]
```

Ausgabe von **`ExecuteBash`** für Iteration 2

```
[{"stdout":"World"}]
```

Ausgabe von **`ExecuteBash`** für Schritt

```
[{"stdout":"Hello\nWorld"}]
```

Beispielsweise `ExecuteBinary` sind `ExecuteBash`, `ExecutePowerShell`und Aktionsmodule, die `STDOUT` als Ausgabe des Aktionsmoduls zurückgeben. -`STDOUT`Nachrichten werden mit dem neuen Zeilenzeichen verknüpft, um die Gesamtausgabe des Schritts in zu erzeugen`detailedOutput.json`.

AWSTOE konsolidiert die Ergebnisse erfolgloser Iterationen nicht.

Vom AWSTOE Komponentenmanager unterstützte Aktionsmodule

Image-Building-Services wie EC2 Image Builder verwenden AWSTOE Aktionsmodule, um die EC2-Instances zu konfigurieren, die zum Erstellen und Testen benutzerdefinierter Computer-Images verwendet werden. In diesem Abschnitt werden die Funktionen häufig verwendeter AWSTOE Aktionsmodule beschrieben und wie sie konfiguriert werden, einschließlich Beispielen.

AWSTOE -Komponenten werden mit Klartext-YAML-Dokumenten erstellt. Weitere Informationen zur Dokumentsyntax finden Sie unter Verwenden von Komponentendokumenten in AWSTOE.

> **Note**
>
> Alle Aktionsmodule verwenden dasselbe Konto wie der Systems Manager-Agent, wenn sie ausgeführt werden, d. h. `root` unter Linux und `NT Authority\SYSTEM` unter Windows.

Aktionsmodultypen

- Allgemeine Ausführungsmodule
- Module zum Herunterladen und Hochladen von Dateien
- Dateisystem-Betriebsmodule
- Softwareinstallationsaktionen
- Module für Systemaktionen

Allgemeine Ausführungsmodule

Der folgende Abschnitt enthält Details zu Aktionsmodulen, die allgemeine Ausführungsbefehle und Anweisungen ausführen.

Allgemeine Module für Ausführungsaktionen

- ExecuteBash
- ExecuteBinary
- ExecuteDocument
- ExecutePowerShell

ExecuteBash

Mit dem ExecuteBash Aktionsmodul können Sie Bash-Skripte mit Inline-Shell-Code/-Befehlen ausführen. Dieses Modul unterstützt Linux.

Alle Befehle und Anweisungen, die Sie im Befehlsblock angeben, werden in eine Datei konvertiert (z. B. `input.sh`) und mit der Bash-Shell ausgeführt. Das Ergebnis der Ausführung der Shell-Datei ist der Beendigungscode des Schritts.

Das ExecuteBash Modul verarbeitet Systemneustarts, wenn das Skript mit dem Beendigungscode beendet wird`194`. Bei der Initiierung führt die Anwendung eine der folgenden Aktionen aus:

- Die Anwendung übergibt den Beendigungscode an den Aufrufer, wenn er vom Systems Manager Agent ausgeführt wird. Der Systems Manager Agent verarbeitet den Systemneustart und führt denselben Schritt aus, der den Neustart initiiert hat, wie unter Neustarten verwalteter Instances von Skripts beschrieben.
- Die Anwendung speichert das aktuelle `executionstate`, konfiguriert einen Neustartauslöser zum erneuten Ausführen der Anwendung und startet das System neu.

Nach dem Systemneustart führt die Anwendung denselben Schritt aus, der den Neustart initiiert hat. Wenn Sie diese Funktionalität benötigen, müssen Sie idempotente Skripte schreiben, die mehrere Aufrufe desselben Shell-Befehls verarbeiten können.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`commands`	Enthält eine Liste von Anweisungen oder Befehlen, die gemäß Bash-Syntax ausgeführt werden sollen. Mehrzeiliges YAML ist zulässig.	Auflisten	Ja

Eingabebeispiel: Vor und nach einem Neustart

```
name: ExitCode194Example
description: This shows how the exit code can be used to restart a system with
 ExecuteBash
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: RestartTrigger
        action: ExecuteBash
        inputs:
          commands:
            - |
              REBOOT_INDICATOR=/var/tmp/reboot-indicator
              if [ -f "${REBOOT_INDICATOR}" ]; then
                echo 'The reboot file exists. Deleting it and exiting with success.'
                rm "${REBOOT_INDICATOR}"
                exit 0
              fi
              echo 'The reboot file does not exist. Creating it and triggering a
 restart.'
              touch "${REBOOT_INDICATOR}"
              exit 194
```

Output

Feld	Beschreibung	Typ
`stdout`	Standardausgabe der Befehlsausführung.	Zeichenfolge

Wenn Sie einen Neustart starten und Beendigungscode `194` als Teil des Aktionsmoduls zurückgeben, wird der Build mit demselben Schritt des Aktionsmoduls fortgesetzt, der den Neustart initiiert hat. Wenn Sie einen Neustart ohne Beendigungscode starten, schlägt der Build-Prozess möglicherweise fehl.

Ausgabebeispiel: Vor dem Neustart (erstmalig über Dokument)

```
{
    “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

Ausgabebeispiel: Nach dem Neustart (Sekundenmalig bis -Dokument)

```
{
    “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

ExecuteBinary

Mit dem ExecuteBinary Aktionsmodul können Sie Binärdateien mit einer Liste von Befehlszeilenargumenten ausführen.

Das ExecuteBinary Modul verarbeitet Systemneustarts, wenn die Binärdatei mit dem Beendigungscode `194` (Linux) oder `3010` (Windows) beendet wird. In diesem Fall führt die Anwendung eine der folgenden Aktionen aus:

- Die Anwendung übergibt den Beendigungscode an den Aufrufer, wenn er vom Systems Manager Agent ausgeführt wird. Der Systems Manager Agent führt den Neustart des Systems durch und führt denselben Schritt aus, der den Neustart initiiert hat, wie unter Neustarten einer verwalteten Instance von Skripts beschrieben.
- Die Anwendung speichert das aktuelle `executionstate`, konfiguriert einen Neustartauslöser zum erneuten Ausführen der Anwendung und startet das System neu.

Nach dem Neustart des Systems führt die Anwendung denselben Schritt aus, der den Neustart initiiert hat. Wenn Sie diese Funktionalität benötigen, müssen Sie idempotente Skripte schreiben, die mehrere Aufrufe desselben Shell-Befehls verarbeiten können.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`path`	Der Pfad zur Binärdatei zur Ausführung.	String	Ja
`arguments`	Enthält eine Liste von Befehlsze ilenargumenten, die beim Ausführen der Binärdatei verwendet werden sollen.	Zeichenfolgenliste	Nein

Eingabebeispiel: Installieren von .NET

```
name: "InstallDotnet"
action: ExecuteBinary
inputs:
  path: C:\PathTo\dotnet_installer.exe
  arguments:
    - /qb
    - /norestart
```

Output

Feld	Beschreibung	Typ
`stdout`	Standardausgabe der Befehlsausführung.	Zeichenfolge

Output example

```
{
```

```
      "stdout": "success"
}
```

ExecuteDocument

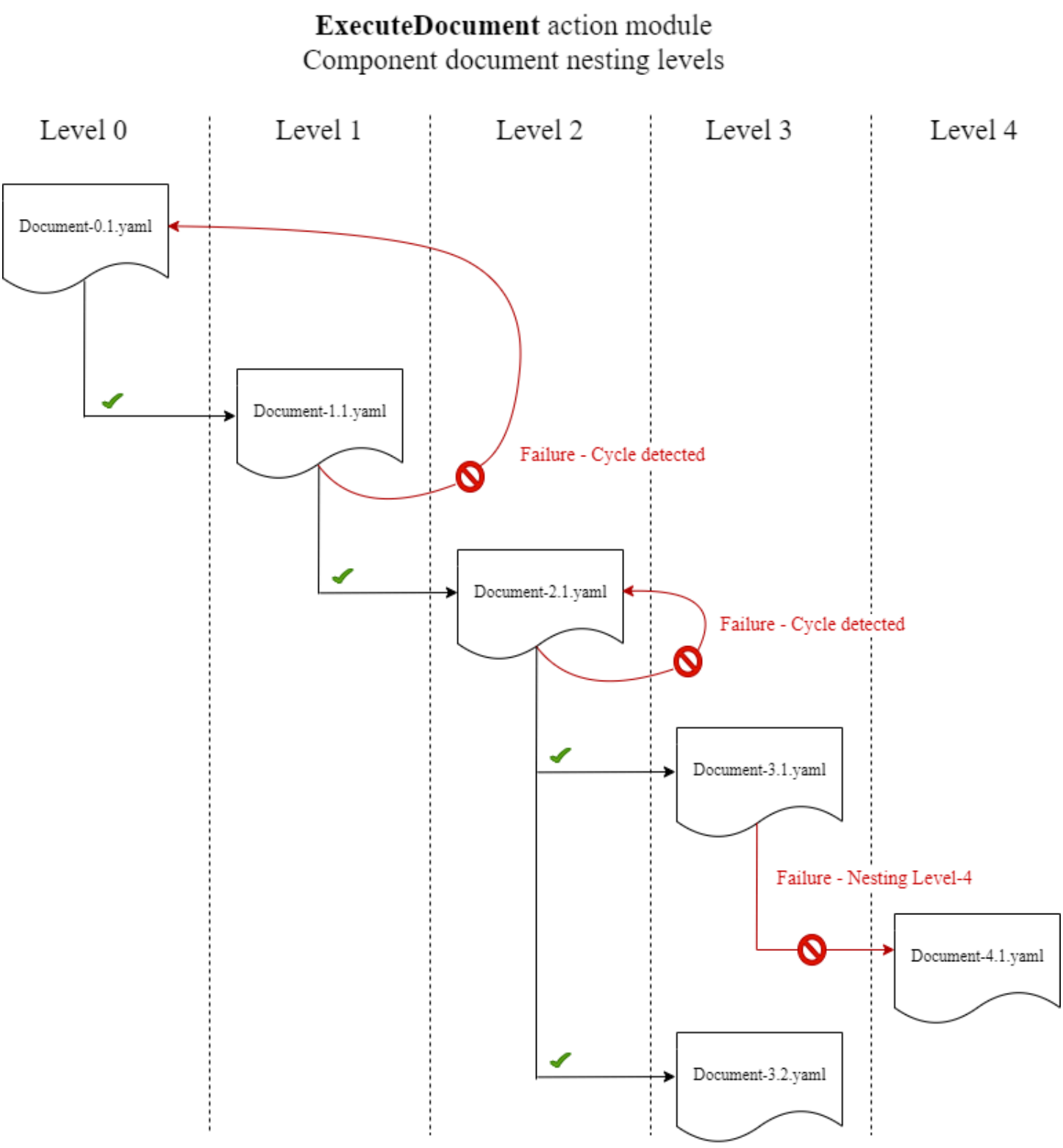
Das ExecuteDocument Aktionsmodul bietet Unterstützung für verschachtelte Komponentendokumente und führt mehrere Komponentendokumente aus einem Dokument aus. AWSTOE validiert das Dokument, das zur Laufzeit im Eingabeparameter übergeben wird.

Einschränkungen

- Dieses Aktionsmodul wird einmal ausgeführt, ohne dass Wiederholungsversuche zulässig sind und keine Option zum Festlegen von Timeout-Limits besteht. ExecuteDocument legt die folgenden Standardwerte fest und gibt einen Fehler zurück, wenn Sie versuchen, sie zu ändern.
  - `timeoutSeconds`: -1
  - `maxAttempts`: 1

  > **Note**
  >
  > Sie können diese Werte leer lassen und AWSTOE verwendet die Standardwerte.

- Die Dokumentenverschachtelung ist zulässig, bis zu drei Ebenen tief, aber nicht mehr. Drei Verschachtelungsebenen werden in vier Dokumentebenen übersetzt, da die oberste Ebene nicht verschachtelt ist. In diesem Szenario darf das Dokument der untersten Ebene keine anderen Dokumente aufrufen.
- Die zyklische Ausführung von Komponentendokumenten ist nicht zulässig. Jedes Dokument, das sich außerhalb eines Schleifenkonstrukts aufruft oder ein anderes Dokument höher in der aktuellen Ausführungskette aufruft, initiiert einen Zyklus, der zu einer Endlosschleife führen kann. Wenn eine zyklische Ausführung AWSTOE erkennt, wird die Ausführung gestoppt und der Fehler aufgezeichnet.



Wenn ein Komponentendokument versucht, sich selbst oder eines der Komponentendokumente auszuführen, die in der aktuellen Ausführungskette höher sind, schlägt die Ausführung fehl.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`document`	Pfad des Komponent endokuments. Gültige Optionen sind unter anderem: • Lokale Dateipfade • S3-URIs • EC2 Image Builder-Komponenten-Build-Versions-ARNs	String	Ja
`document-s3-bucket-owner`	Die Konto-ID des S3-Bucket-Eigentümers für den S3-Bucket, in dem Komponent endokumente gespeichert sind. (Empfohlen, wenn Sie S3-URIs in Ihrem Komponent endokument verwenden.)	String	Nein
`phases`	Phasen, die im Komponent endokument ausgeführt werden sollen, ausgedrückt als kommagetrennte Liste. Wenn keine Phasen angegeben	String	Nein

Primitiv	Beschreibung	Typ	Erforderlich
	sind, werden alle Phasen ausgeführt.		
`parameters`	Eingabeparameter, die zur Laufzeit als Schlüssel-Wert-Paare an das Komponentendokument übergeben werden.	Liste der Parameterzuordnungen	Nein

Eingabe der Parameterzuordnung

Primitiv	Beschreibung	Typ	Erforderlich
`name`	Der Name des Eingabeparameters, der an das Komponentendokument übergeben werden soll, das das ExecuteDocument Aktionsmodul ausführt.	String	Ja
`value`	Der Wert des Eingabeparameters.	String	Ja

Eingabebeispiele

Die folgenden Beispiele zeigen je nach Installationspfad Variationen der Eingaben für Ihr Komponentendokument.

Eingabebeispiel: Lokaler Dokumentpfad

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: Sample-1.yaml
          phases: build
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Eingabebeispiel: S3-URI als Dokumentpfad

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: s3://my-bucket/Sample-1.yaml
          document-s3-bucket-owner: 123456789012
          phases: build,validate
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Eingabebeispiel: EC2 Image Builder-Komponenten-ARN als Dokumentpfad

```
# main.yaml
schemaVersion: 1.0
```

```
phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: arn:aws:imagebuilder:us-west-2:aws:component/Sample-Test/1.0.0
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Verwenden einer ForEach Schleife zum Ausführen von Dokumenten

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForEachLoop'
          forEach:
            - Sample-1.yaml
            - Sample-2.yaml
        inputs:
          document: "{{myForEachLoop.value}}"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Verwenden einer For-Loop zum Ausführen von Dokumenten

```
# main.yaml
schemaVersion: 1.0
```

```
phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForLoop'
          for:
            start: 1
            end: 2
            updateBy: 1
        inputs:
          document: "Sample-{{myForLoop.value}}.yaml"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Output

AWSTOE erstellt bei `detailedoutput.json` jeder Ausführung eine Ausgabedatei mit dem Namen . Die Datei enthält Details zu jeder Phase und jedem Schritt jedes Komponentendokuments, das während der Ausführung aufgerufen wird. Für das ExecuteDocument Aktionsmodul finden Sie eine kurze Laufzeitübersicht im `outputs` Feld sowie Details zu den Phasen, Schritten und Dokumenten, die es in der ausführt`detailedOutput`.

```
"outputs": "[{\"executedStepCount\":1,\"executionId\":\"97054e22-06cc-11ec-9b14-
acde48001122\",\"failedStepCount\":0,\"failureMessage\":\"\",\"ignoredFailedStepCount
\":0,\"logUrl\":\"\",\"status\":\"success\"}]",
```

Das Ausgabeübersichtsobjekt jedes Komponentendokuments enthält die folgenden Details mit Beispielwerten, wie hier gezeigt:

- executedStepCount„:1
- "executionId ":"12345a67-89bc-01de-2f34-abcd56789012"
- "failedStepCount":0
- "failureMessage ":""
- „ignoredFailedStepAnzahl“:0

- "logUrl ":""
- „Status“: „Erfolg“

Ausgabebeispiel

Das folgende Beispiel zeigt die Ausgabe des -ExecuteDocumentAktionsmoduls, wenn eine verschachtelte Ausführung stattfindet. In diesem Beispiel führt das `main.yaml` Komponentendokument das `Sample-1.yaml` Komponentendokument erfolgreich aus.


```
{
    "executionId": "12345a67-89bc-01de-2f34-abcd56789012",
    "status": "success",
    "startTime": "2021-08-26T17:20:31-07:00",
    "endTime": "2021-08-26T17:20:31-07:00",
    "failureMessage": "",
    "documents": [
        {
            "name": "",
            "filePath": "main.yaml",
            "status": "success",
            "description": "",
            "startTime": "2021-08-26T17:20:31-07:00",
            "endTime": "2021-08-26T17:20:31-07:00",
            "failureMessage": "",
            "phases": [
                {
                    "name": "build",
                    "status": "success",
                    "startTime": "2021-08-26T17:20:31-07:00",
                    "endTime": "2021-08-26T17:20:31-07:00",
                    "failureMessage": "",
                    "steps": [
                        {
                            "name": "ExecuteNestedDocument",
                            "status": "success",
                            "failureMessage": "",
                            "timeoutSeconds": -1,
                            "onFailure": "Abort",
                            "maxAttempts": 1,
                            "action": "ExecuteDocument",
                            "startTime": "2021-08-26T17:20:31-07:00",
                            "endTime": "2021-08-26T17:20:31-07:00",
```

```
                            "inputs": "[{\"document\":\"Sample-1.yaml\",\"document-s3-
bucket-owner\":\"\",\"phases\":\"\",\"parameters\":null}]",
                            "outputs": "[{\"executedStepCount\":1,\"executionId\":
\"98765f43-21ed-09cb-8a76-fedc54321098\",\"failedStepCount\":0,\"failureMessage\":\"\",
\"ignoredFailedStepCount\":0,\"logUrl\":\"\",\"status\":\"success\"}]",
                            "loop": null,
                            "detailedOutput": [
                                {
                                    "executionId": "98765f43-21ed-09cb-8a76-
fedc54321098",
                                    "status": "success",
                                    "startTime": "2021-08-26T17:20:31-07:00",
                                    "endTime": "2021-08-26T17:20:31-07:00",
                                    "failureMessage": "",
                                    "documents": [
                                        {
                                            "name": "",
                                            "filePath": "Sample-1.yaml",
                                            "status": "success",
                                            "description": "",
                                            "startTime": "2021-08-26T17:20:31-07:00",
                                            "endTime": "2021-08-26T17:20:31-07:00",
                                            "failureMessage": "",
                                            "phases": [
                                                {
                                                    "name": "build",
                                                    "status": "success",
                                                    "startTime":
 "2021-08-26T17:20:31-07:00",
                                                    "endTime":
 "2021-08-26T17:20:31-07:00",
                                                    "failureMessage": "",
                                                    "steps": [
                                                        {
                                                            "name": "ExecuteBashStep",
                                                            "status": "success",
                                                            "failureMessage": "",
                                                            "timeoutSeconds": 7200,
                                                            "onFailure": "Abort",
                                                            "maxAttempts": 1,
                                                            "action": "ExecuteBash",
                                                            "startTime":
 "2021-08-26T17:20:31-07:00",
```

```
                                                                "endTime":
 "2021-08-26T17:20:31-07:00",
                                                                "inputs": "[{\"commands\":
[\"echo \\\"Hello World!\\\"\"]}]",
                                                                "outputs": "[{\"stdout\":
\"Hello World!\"}]",
                                                                "loop": null,
                                                                "detailedOutput": null
                                                            }]
                                                        }]
                                                    }]
                                                }]
                                            }]

                                    }]
                                }]
}
```

ExecutePowerShell

Mit dem ExecutePowerShell Aktionsmodul können Sie PowerShell Skripts mit Inline-Shell-Code/Befehlen ausführen. Dieses Modul unterstützt die Windows-Plattform und Windows PowerShell.

Alle im Befehlsblock angegebenen Befehle/Anweisungen werden in eine Skriptdatei konvertiert (z. B. `input.ps1`) und mit Windows ausgeführtPowerShell. Das Ergebnis der Ausführung der Shell-Datei ist der Beendigungscode.

Das ExecutePowerShell Modul verarbeitet Systemneustarts, wenn der Shell-Befehl mit dem Beendigungscode beendet wird`3010`. Bei der Initiierung führt die Anwendung eine der folgenden Aktionen aus:

- Übergibt den Beendigungscode an den Aufrufer, wenn er vom Systems Manager Agent ausgeführt wird. Der Systems Manager Agent verarbeitet den Systemneustart und führt denselben Schritt aus, der den Neustart initiiert hat, wie unter Neustarten verwalteter Instances von Skripts beschrieben.
- Speichert das aktuelle `executionstate`, konfiguriert einen Neustartauslöser zum erneuten Ausführen der Anwendung und startet das System neu.

Nach dem Systemneustart führt die Anwendung denselben Schritt aus, der den Neustart initiiert hat. Wenn Sie diese Funktionalität benötigen, müssen Sie idempotente Skripte schreiben, die mehrere Aufrufe desselben Shell-Befehls verarbeiten können.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`commands`	Enthält eine Liste von Anweisungen oder Befehlen, die gemäß der PowerShell Syntax ausgeführt werden sollen. Mehrzeiliges YAML ist zulässig.	Zeichenfolgenliste	Ja. Muss `commands` oder angeben`file`, nicht beides.
`file`	Enthält den Pfad zu einer PowerShell Skriptdatei. PowerShell wird mit dem `-file` Befehlszeilenargument für diese Datei ausgeführt. Der Pfad muss auf eine `.ps1` Datei verweisen.	String	Ja. Muss `commands` oder angeben`file`, nicht beides.

Eingabebeispiel: Vor und nach einem Neustart

```
name: ExitCode3010Example
description: This shows how the exit code can be used to restart a system with
 ExecutePowerShell
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: RestartTrigger
        action: ExecutePowerShell
        inputs:
          commands:
            - |
```

```
              $rebootIndicator = Join-Path -Path $env:SystemDrive -ChildPath 'reboot-
indicator'
              if (Test-Path -Path $rebootIndicator) {
                Write-Host 'The reboot file exists. Deleting it and exiting with
 success.'
                Remove-Item -Path $rebootIndicator -Force | Out-Null
                [System.Environment]::Exit(0)
              }
              Write-Host 'The reboot file does not exist. Creating it and triggering a
 restart.'
              New-Item -Path $rebootIndicator -ItemType File | Out-Null
              [System.Environment]::Exit(3010)
```

Output

Feld	Beschreibung	Typ
`stdout`	Standardausgabe der Befehlsausführung.	Zeichenfolge

Wenn Sie `3010` im Rahmen des Aktionsmoduls einen Neustart ausführen und Beendigungscode zurückgeben, wird der Build mit demselben Schritt des Aktionsmoduls fortgesetzt, der den Neustart initiiert hat. Wenn Sie einen Neustart ohne Beendigungscode ausführen, schlägt der Build-Prozess möglicherweise fehl.

Ausgabebeispiel: Vor dem Neustart (erstmalig über Dokument)

```
{
    “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

Ausgabebeispiel: Nach dem Neustart (Sekundenmalig bis -Dokument)

```
{
    “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

Module zum Herunterladen und Hochladen von Dateien

Der folgende Abschnitt enthält Details zu Aktionsmodulen, die Download- und Upload-Befehle und -Anweisungen ausführen.

Herunterladen und Hochladen von Aktionsmodulen

- S3Download
- S3Upload
- WebDownload

S3Download

Mit dem `S3Download` Aktionsmodul können Sie ein Amazon S3-Objekt oder eine Reihe von Objekten in eine lokale Datei oder einen lokalen Ordner herunterladen, die/den Sie mit dem `destination` Pfad angeben. Wenn am angegebenen Speicherort bereits eine Datei vorhanden ist und das `overwrite` Flag auf „true“ gesetzt ist, `S3Download` überschreibt die Datei.

Ihr `source` Speicherort kann auf ein bestimmtes Objekt in Amazon S3 verweisen, oder Sie können ein Schlüsselpräfix mit einem Sternchen-Platzhalter (*) verwenden, um eine Reihe von Objekten herunterzuladen, die dem Schlüsselpräfixpfad entsprechen. Wenn Sie ein Schlüsselpräfix an Ihrem `source` Speicherort angeben, lädt das `S3Download` Aktionsmodul alles herunter, das dem Präfix entspricht (Dateien und Ordner enthalten). Stellen Sie sicher, dass das Schlüsselpräfix mit einem Vorwärtsschrägstrich endet, gefolgt von einem Sternchen (/*), sodass Sie alles herunterladen, das dem Präfix entspricht. Zum Beispiel: *`s3://my-bucket/my-folder/*`*.

> **Note**
>
> Alle Ordner im Zielpfad müssen vor dem Download vorhanden sein, sonst schlägt der Download fehl.

Wenn die `S3Download` Aktion für ein bestimmtes Schlüsselpräfix während eines Downloads fehlschlägt, wird der Ordnerinhalt nicht in seinen Zustand zurückgesetzt, bevor der Fehler auftritt. Der Zielordner bleibt so, wie er zum Zeitpunkt des Fehlers war.

Unterstützte Anwendungsfälle

Das `S3Download` Aktionsmodul unterstützt die folgenden Anwendungsfälle:

- Das Amazon S3-Objekt wird in einen lokalen Ordner heruntergeladen, wie im Download-Pfad angegeben.
- Amazon S3-Objekte (mit einem Schlüsselpräfix im Amazon S3-Dateipfad) werden in den angegebenen lokalen Ordner heruntergeladen, der rekursiv alle Amazon S3-Objekte kopiert, die dem Schlüsselpräfix entsprechen, in den lokalen Ordner.

IAM-Anforderungen

Die IAM-Rolle, die Sie Ihrem Instance-Profil zuordnen, muss über Berechtigungen zum Ausführen des `S3Download` Aktionsmoduls verfügen. Die folgenden IAM-Richtlinien müssen an die IAM-Rolle angehängt werden, die dem Instance-Profil zugeordnet ist:

- Einzelne Datei : `s3:GetObject` gegen den Bucket/das Objekt (z. B. `arn:aws:s3:::BucketName/*`).
- Mehrere Dateien : `s3:ListBucket` für den Bucket/das Objekt (z. B. `arn:aws:s3:::BucketName`) und `s3:GetObject` für den Bucket/das Objekt (z. B. `arn:aws:s3:::BucketName/*`).

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`source`	Der Amazon S3-Bucket, der die Quelle für Ihren Download ist. Sie können einen Pfad zu einem bestimmten Objekt angeben oder ein Schlüsselpräfix verwenden, das mit einem Vorwärtsschrägstrich endet, gefolgt	String	Ja	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	von einem Sternchen-Platzhalter (`/*`), um eine Reihe von Objekten herunterz uladen, die dem Schlüsselpräfix entsprechen.			
`destination`	Der lokale Pfad, in den die Amazon S3-Objekte heruntergeladen werden. Um eine einzelne Datei herunterzuladen, müssen Sie den Dateiname n als Teil des Pfads angeben. Beispiel: */myfolder/package.zip*	String	Ja	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`expectedBucketOwner`	Erwartete Besitzerkonto-ID des Buckets, der im `source` Pfad angegeben ist. Wir empfehlen Ihnen, den Besitz des in der Quelle angegebenen Amazon S3-Buckets zu überprüfen.	String	Nein	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`overwrite`	Wenn auf „true“ gesetzt und bereits eine Datei mit demselben Namen im Zielordner für den angegeben en lokalen Pfad vorhanden ist, überschreibt die Download-Datei die lokale Datei. Wenn diese Option auf „false“ gesetzt ist, wird die vorhanden e Datei auf dem lokalen System vor dem Überschreiben geschützt und das Aktionsmo dul schlägt mit einem Download-Fehler fehl. Beispiel: `Error: S3Download: File already exists and "overwrit e" prope rty for`	Boolesch	Nein	true

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	"destination" file is set to false. Cannot download.			

> **Note**
>
> In den folgenden Beispielen kann der Windows-Ordnerpfad durch einen Linux-Pfad ersetzt werden. Beispielsweise *C:\myfolder\package.zip* kann durch ersetzt werden*/myfolder/package.zip*.

Eingabebeispiel: Kopieren eines Amazon S3-Objekts in eine lokale Datei

Das folgende Beispiel zeigt, wie Sie ein Amazon S3-Objekt in eine lokale Datei kopieren.

```
name: DownloadMyFile
action: S3Download
inputs:
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: true
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
```

Eingabebeispiel: Kopieren aller Amazon S3-Objekte in einem Amazon S3-Bucket mit Schlüsselpräfix in einen lokalen Ordner

Das folgende Beispiel zeigt, wie Sie alle Amazon S3-Objekte in einem Amazon S3-Bucket mit dem Schlüsselpräfix in einen lokalen Ordner kopieren. Amazon S3 hat kein Ordnerkonzept, daher werden

alle Objekte kopiert, die mit dem Schlüsselpräfix übereinstimmen. Die maximale Anzahl der Objekte, die heruntergeladen werden können, beträgt 1 000.

```
name: MyS3DownloadKeyprefix
action: S3Download
maxAttempts: 3
inputs:
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
      overwrite: true
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
```

Output

Keine.

S3Upload

Mit dem S3Upload-Aktionsmodul können Sie eine Datei aus einer Quelldatei oder einem Ordner an einen Amazon S3-Speicherort hochladen. Sie können einen Platzhalter (*) in dem Pfad verwenden, der für Ihren Quellspeicherort angegeben ist, um alle Dateien hochzuladen, deren Pfad mit dem Platzhaltermuster übereinstimmt.

Wenn die rekursive S3Upload-Aktion fehlschlägt, verbleiben alle Dateien, die bereits hochgeladen wurden, im Amazon S3-Ziel-Bucket.

Unterstützte Anwendungsfälle

- Lokale Datei im Amazon S3-Objekt.
- Lokale Dateien im Ordner (mit Platzhalter) zu Amazon S3-Schlüsselpräfix.
- Kopieren Sie den lokalen Ordner (muss auf `recurse` gesetzt sein`true`) in das Amazon S3-Schlüsselpräfix.

IAM-Anforderungen

Die IAM-Rolle, die Sie Ihrem Instance-Profil zuordnen, muss über Berechtigungen zum Ausführen des `S3Upload` Aktionsmoduls verfügen. Die folgende IAM-Richtlinie muss an die IAM-Rolle angehängt werden, die dem Instance-Profil zugeordnet ist. Die Richtlinie muss dem Amazon S3-Ziel-Bucket `s3:PutObject` Berechtigungen erteilen. Beispiel, `arn:aws:s3:::BucketName/*`).

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`source`	Der lokale Pfad, aus dem Quelldateien/-ordner stammen. unterstützt `source` einen Sternchen-Platzhalter (`*`).	String	Ja	N/A
`destination`	Der Pfad für den Amazon S3-Ziel-Bucket, in den Quelldateien/-ordner hochgeladen werden.	String	Ja	N/A
`recurse`	Wenn auf festgelegt`true`, führt S3Upload rekursiv aus.	String	Nein	`false`
`expectedBucketOwner`	Die erwartete Besitzerkonto-ID für den Amazon S3-Bucket, der im Zielpfad angegeben ist. Wir empfehlen Ihnen, den	String	Nein	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	Besitz des Amazon S3-Buckets zu überprüfen, der im Ziel angegeben ist.			

Eingabebeispiel: Kopieren einer lokalen Datei in ein Amazon S3-Objekt

Das folgende Beispiel zeigt, wie Sie eine lokale Datei in ein Amazon S3-Objekt kopieren.

```
name: MyS3UploadFile
action: S3Upload
onFailure: Abort
maxAttempts: 3
inputs:
    - source: C:\myfolder\package.zip
      destination: s3://mybucket/path/to/package.zip
      expectedBucketOwner: 123456789022
```

Eingabebeispiel: Kopieren aller Dateien in einem lokalen Ordner in einen Amazon S3-Bucket mit Schlüsselpräfix

Das folgende Beispiel zeigt, wie Sie alle Dateien im lokalen Ordner in einen Amazon S3-Bucket mit dem Schlüsselpräfix kopieren. In diesem Beispiel werden Unterordner oder deren Inhalt nicht kopiert, da nicht angegeben `recurse` ist und es standardmäßig ist`false`.

```
name: MyS3UploadMultipleFiles
action: S3Upload
onFailure: Abort
maxAttempts: 3
inputs:
    - source: C:\myfolder\*
      destination: s3://mybucket/path/to/
      expectedBucketOwner: 123456789022
```

Eingabebeispiel: Kopieren Sie alle Dateien und Ordner rekursiv aus einem lokalen Ordner in einen Amazon S3-Bucket

Das folgende Beispiel zeigt, wie Sie alle Dateien und Ordner rekursiv aus einem lokalen Ordner in einen Amazon S3-Bucket mit dem Schlüsselpräfix kopieren.

```
name: MyS3UploadFolder
action: S3Upload
onFailure: Abort
maxAttempts: 3
inputs:
    - source: C:\myfolder\*
      destination: s3://mybucket/path/to/
      recurse: true
      expectedBucketOwner: 123456789022
```

Output

Keine.

WebDownload

Mit dem WebDownload Aktionsmodul können Sie Dateien und Ressourcen von einem Remote-Speicherort über das HTTP/HTTPS-Protokoll herunterladen (HTTPS wird empfohlen). Die Anzahl oder Größe der Downloads ist unbegrenzt. Dieses Modul verarbeitet die Logik für Wiederholungsversuche und exponentielle Backoffs.

Jedem Download-Vorgang werden maximal 5 Versuche zugewiesen, gemäß den Benutzereingaben erfolgreich zu sein. Diese Versuche unterscheiden sich von denen, die im `-maxAttempts`Feld des Dokuments angegeben sind`steps`, die sich auf Fehler bei Aktionsmodulen beziehen.

Dieses Aktionsmodul verarbeitet implizit Umleitungen. Alle HTTP-Statuscodes, mit Ausnahme von `200`, führen zu einem Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`source`	Die gültige HTTP/HTTPS-URL (HTTPS) wird empfohlen , die dem RFC 3986-Standard	String	Ja	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	folgt. Verkettungsausdrücke sind zulässig.			
`destination`	Eine absolute oder relative Datei oder ein Ordnerpfad auf dem lokalen System. Ordnerpfade müssen mit enden`/`. Wenn sie nicht mit enden`/`, werden sie als Dateipfade behandelt. Das Modul erstellt alle erforderlichen Dateien oder Ordner für erfolgreiche Downloads. Verkettungsausdrücke sind zulässig.	String	Ja	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`overwrite`	Wenn diese Option aktiviert ist, überschreibt alle vorhandenen Dateien auf dem lokalen System mit der heruntergeladenen Datei oder Ressource. Wenn diese Option nicht aktiviert ist, werden alle vorhandenen Dateien auf dem lokalen System nicht überschrieben und das Aktionsmodul schlägt mit einem Fehler fehl. Wenn Überschreiben aktiviert ist und Prüfsumme und Algorithmus angegeben sind, lädt das Aktionsmodul die Datei nur herunter, wenn die Prüfsumme und der Hash der bereits	Boolesch	Nein	`true`

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	vorhandenen Dateien nicht übereinstimmen.			
`checksum`	Wenn Sie die Prüfsumme angeben, wird sie mit dem Hash der heruntergeladenen Datei verglichen, die mit dem bereitgestellten Algorithmus generiert wird. Damit die Dateiverifizierung aktiviert werden kann, müssen sowohl die Prüfsumme als auch der Algorithmus bereitgestellt werden. Verkettungsausdrücke sind zulässig.	String	Nein	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`algorithm`	Der Algorithm us, der zur Berechnung der Prüfsumme verwendet wird. Die Optionen sind MD5, SHA1, SHA256 und SHA512. Damit die Dateiveri fizierung aktiviert werden kann, müssen sowohl die Prüfsumme als auch der Algorithm us bereitges tellt werden. Verkettun gsausdrücke sind zulässig.	String	Nein	N/A
`ignoreCer tificateE rrors`	Die SSL-Zerti fikatvalidierung wird ignoriert, wenn sie aktiviert ist.	Boolesch	Nein	`false`

Output

Primitiv	Beschreib ung	Typ				
`destinati on`	Zeichenfo lge mit	String				

Primitiv	Beschreibung	Typ				
	Zeilenumbruchzeichen als Trennzeichen, die den Zielpfad angibt, in dem die heruntergeladenen Dateien oder Ressourcen gespeichert werden.					

Eingabebeispiel: Remote-Datei auf das lokale Ziel herunterladen

```
name: DownloadRemoteFile
action: WebDownload
maxAttempts: 3
inputs:
  - source: https://testdomain/path/to/java14.zip
    destination: C:\testfolder\package.zip


Output:
{
  "destination": "C:\\testfolder\\package.zip"
}
```

Eingabebeispiel: Herunterladen von mehr als einer Remote-Datei zu mehr als einem lokalen Ziel

```
name: DownloadRemoteFiles
action: WebDownload
maxAttempts: 3
inputs:
  - source: https://testdomain/path/to/java14.zip
    destination: /tmp/java14_renamed.zip
  - source: https://testdomain/path/to/java14.zip
    destination: /tmp/create_new_folder_and_add_java14_as_zip/


Output:
{
  "destination": "/tmp/create_new_folder/java14_renamed.zip\n/tmp/
create_new_folder_and_add_java14_as_zip/java14.zip"
}
```

Eingabebeispiel: Laden Sie eine Remote-Datei herunter, ohne das lokale Ziel zu überschreiben, und laden Sie eine weitere Remote-Datei mit Dateiverifizierung herunter

```
name: DownloadRemoteMultipleProperties
action: WebDownload
maxAttempts: 3
inputs:
  - source: https://testdomain/path/to/java14.zip
    destination: C:\create_new_folder\java14_renamed.zip
    overwrite: false
  - source: https://testdomain/path/to/java14.zip
    destination: C:\create_new_folder_and_add_java14_as_zip\
    checksum: ac68bbf921d953d1cfab916cb6120864
    algorithm: MD5
    overwrite: true


Output:
{
  "destination": "C:\\create_new_folder\\java14_renamed.zip\nC:\
\create_new_folder_and_add_java14_as_zip\\java14.zip"
}
```

Eingabebeispiel: Laden Sie die Remote-Datei herunter und ignorieren Sie die SSL-Zertifizierungsvalidierung

```
name: DownloadRemoteIgnoreValidation
action: WebDownload
maxAttempts: 3
inputs:
  - source: https://www.bad-ssl.com/resource
    destination: /tmp/downloads/
    ignoreCertificateErrors: true


Output:
{
  "destination": "/tmp/downloads/resource"
}
```

Dateisystem-Betriebsmodule

Der folgende Abschnitt enthält Details zu Aktionsmodulen, die Befehle und Anweisungen zur Dateisystemoperation ausführen.

Dateisystem-Aktionsmodule

- AppendFile
- CopyFile
- CopyFolder
- CreateFile
- CreateFolder
- CreateSymlink
- DeleteFile
- DeleteFolder
- ListFiles
- MoveFile
- MoveFolder
- ReadFile
- SetFileEncoding
- SetFileOwner

- SetFolderOwner
- SetFilePermissions
- SetFolderPermissions

AppendFile

Das AppendFile Aktionsmodul fügt dem bereits vorhandenen Inhalt einer Datei angegebenen Inhalt hinzu.

Wenn sich der Wert der Dateikodierung vom Wert der Standardkodierung (`utf-8`) unterscheidet, können Sie den Wert der Dateikodierung mithilfe der `encoding` Option angeben. Standardmäßig `utf-32` wird davon ausgegangen, dass `utf-16` und die Little-Endian-Kodierung verwenden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Die angegebene Datei ist zur Laufzeit nicht vorhanden.
- Sie haben keine Schreibberechtigungen zum Ändern des Dateiinhalts.
- Das Modul stößt während des Dateivorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja
`content`	Der Inhalt, der an die Datei angehängt werden soll.	String	Nein	Leere Zeichenfolge	N/A	Ja
`encoding`	Der Codierungsstandard.	String	Nein	`utf8`	`utf8`, `utf-8`,	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
					`utf16utf-1` `utf16-LE`, `utf-16-LE` `utf16-BE`, `utf-16-BE` , `utf-32`, `utf32`, `utf32-LEutf-32-LE` , `utf32-BE`, und `utf-32-BE` . Bei dem Wert der Codierungsoption wird die Groß- und Kleinschreibung nicht beachtet.	

Eingabebeispiel: Datei ohne Codierung anhängen (Linux)

```
name: AppendingFileWithOutEncodingLinux
action: AppendFile
inputs:
  - path: ./Sample.txt
```

```
    content: "The string to be appended to the file"
```

Eingabebeispiel: Datei ohne Codierung anhängen (Windows)

```
name: AppendingFileWithOutEncodingWindows
action: AppendFile
inputs:
  - path: C:\MyFolder\MyFile.txt
    content: "The string to be appended to the file"
```

Eingabebeispiel: Datei mit Codierung anhängen (Linux)

```
name: AppendingFileWithEncodingLinux
action: AppendFile
inputs:
  - path: /FolderName/SampleFile.txt
    content: "The string to be appended to the file"
    encoding: UTF-32
```

Eingabebeispiel: Datei mit Codierung anhängen (Windows)

```
name: AppendingFileWithEncodingWindows
action: AppendFile
inputs:
  - path: C:\MyFolderName\SampleFile.txt
    content: "The string to be appended to the file"
    encoding: UTF-32
```

Eingabebeispiel: Datei mit leerer Zeichenfolge anhängen (Linux)

```
name: AppendingEmptyStringLinux
action: AppendFile
inputs:
  - path: /FolderName/SampleFile.txt
```

Eingabebeispiel: Datei mit leerer Zeichenfolge anhängen (Windows)

```
name: AppendingEmptyStringWindows
action: AppendFile
inputs:
```

```
  - path: C:\MyFolderName\SampleFile.txt
```

Output

Keine.

CopyFile

Das CopyFile Aktionsmodul kopiert Dateien aus der angegebenen Quelle in das angegebene Ziel. Standardmäßig erstellt das Modul rekursiv den Zielordner, wenn er zur Laufzeit nicht vorhanden ist.

Wenn eine Datei mit dem angegebenen Namen bereits im angegebenen Ordner vorhanden ist, überschreibt das Aktionsmodul standardmäßig die vorhandene Datei. Sie können dieses Standardverhalten überschreiben, indem Sie die Überschreiboption auf setzen`false`. Wenn die Überschreiboption auf festgelegt ist `false`und bereits eine Datei am angegebenen Speicherort mit dem angegebenen Namen vorhanden ist, gibt das Aktionsmodul einen Fehler zurück. Diese Option funktioniert genauso wie der `cp` Befehl unter Linux, der standardmäßig überschreibt.

Der Quelldateiname kann einen Platzhalter (`*`) enthalten. Platzhalterzeichen werden erst nach dem letzten Dateipfadtrennzeichen (`/` oder `\`) akzeptiert. Wenn Platzhalterzeichen im Quelldateinamen enthalten sind, werden alle Dateien, die mit dem Platzhalter übereinstimmen, in den Zielordner kopiert. Wenn Sie mehr als eine Datei mithilfe eines Platzhalterzeichens verschieben möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (`/` oder `\`) enden, das angibt, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Zieldateiname vom Quelldateinamen unterscheidet, können Sie den Zieldateinamen mit der `destination` Option angeben. Wenn Sie keinen Zieldateinamen angeben, wird der Name der Quelldatei verwendet, um die Zieldatei zu erstellen. Jeder Text, der dem letzten Dateipfadtrennzeichen (`/` oder `\`) folgt, wird als Dateiname behandelt. Wenn Sie denselben Dateinamen wie die Quelldatei verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (`/` oder `\`) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, eine Datei im angegebenen Ordner zu erstellen.
- Die Quelldateien sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Dateinamen und die `overwrite` Option ist auf gesetzt`false`.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`source`	Der Pfad der Quelldatei.	String	Ja	–	N/A	Ja
`destination`	Der Pfad der Zieldatei.	String	Ja	–	N/A	Ja
`overwrite`	Wenn auf „false“ gesetzt, werden die Zieldateien nicht ersetzt, wenn sich bereits eine Datei am angegebenen Speicherort mit dem angegebenen Namen befindet.	Boolesch	Nein	`true`	N/A	Ja

Eingabebeispiel: Kopieren einer Datei (Linux)

```
name: CopyingAFileLinux
action: CopyFile
inputs:
```

```
  - source: /Sample/MyFolder/Sample.txt
    destination: /MyFolder/destinationFile.txt
```

Eingabebeispiel: Kopieren einer Datei (Windows)

```
name: CopyingAFileWindows
action: CopyFile
inputs:
  - source: C:\MyFolder\Sample.txt
    destination: C:\MyFolder\destinationFile.txt
```

Eingabebeispiel: Kopieren einer Datei mit dem Quelldateinamen (Linux)

```
name: CopyingFileWithSourceFileNameLinux
action: CopyFile
inputs:
  - source: /Sample/MyFolder/Sample.txt
    destination: /MyFolder/
```

Eingabebeispiel: Kopieren einer Datei unter Verwendung des Quelldateinamens (Windows)

```
name: CopyingFileWithSourceFileNameWindows
action: CopyFile
inputs:
  - source: C:\Sample\MyFolder\Sample.txt
    destination: C:\MyFolder\
```

Eingabebeispiel: Kopieren einer Datei mit dem Platzhalterzeichen (Linux)

```
name: CopyingFilesWithWildCardLinux
action: CopyFile
inputs:
  - source: /Sample/MyFolder/Sample*
    destination: /MyFolder/
```

Eingabebeispiel: Kopieren einer Datei mit dem Platzhalterzeichen (Windows)

```
name: CopyingFilesWithWildCardWindows
action: CopyFile
inputs:
  - source: C:\Sample\MyFolder\Sample*
```

```
    destination: C:\MyFolder\
```

Eingabebeispiel: Kopieren einer Datei ohne Überschreiben (Linux)

```
name: CopyingFilesWithoutOverwriteLinux
action: CopyFile
inputs:
  - source: /Sample/MyFolder/Sample.txt
    destination: /MyFolder/destinationFile.txt
    overwrite: false
```

Eingabebeispiel: Kopieren einer Datei ohne Überschreiben (Windows)

```
name: CopyingFilesWithoutOverwriteWindows
action: CopyFile
inputs:
  - source: C:\Sample\MyFolder\Sample.txt
    destination: C:\MyFolder\destinationFile.txt
    overwrite: false
```

Output

Keine.

CopyFolder

Das CopyFolder Aktionsmodul kopiert einen Ordner aus der angegebenen Quelle in das angegebene Ziel. Die Eingabe für die `source` Option ist der zu kopierende Ordner, und die Eingabe für die `destination` Option ist der Ordner, in den der Inhalt des Quellordners kopiert wird. Standardmäßig erstellt das Modul rekursiv den Zielordner, wenn er zur Laufzeit nicht vorhanden ist.

Wenn bereits ein Ordner mit dem angegebenen Namen im angegebenen Ordner vorhanden ist, überschreibt das Aktionsmodul standardmäßig den vorhandenen Ordner. Sie können dieses Standardverhalten überschreiben, indem Sie die Überschreiboption auf setzen`false`. Wenn die Überschreiboption auf festgelegt ist `false`und bereits ein Ordner am angegebenen Speicherort mit dem angegebenen Namen vorhanden ist, gibt das Aktionsmodul einen Fehler zurück.

Der Name des Quellordners kann einen Platzhalter (*) enthalten. Platzhalterzeichen werden erst nach dem letzten Dateipfadtrennzeichen (/ oder \) akzeptiert. Wenn Platzhalterzeichen im Quellordnernamen enthalten sind, werden alle Ordner, die mit dem Platzhalter übereinstimmen,

in den Zielordner kopiert. Wenn Sie mehr als einen Ordner mit einem Platzhalterzeichen kopieren möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/ oder \) enden, das angibt, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Name des Zielordners vom Namen des Quellordners unterscheidet, können Sie den Namen des Zielordners mit der `destination` Option angeben. Wenn Sie keinen Zielordnernamen angeben, wird der Name des Quellordners verwendet, um den Zielordner zu erstellen. Jeder Text, der dem letzten Dateipfadtrennzeichen (/ oder \) folgt, wird als Ordnername behandelt. Wenn Sie denselben Ordnernamen wie der Quellordner verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/ oder \) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, einen Ordner im angegebenen Ordner zu erstellen.
- Die Quellordner sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Ordnernamen und die `overwrite` Option ist auf gesetzt`false`.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`source`	Der Quellordnerpfad.	String	Ja	–	N/A	Ja
`destination`	Der Pfad des Zielordners.	String	Ja	–	N/A	Ja
`overwrite`	Wenn der Wert auf „false“ gesetzt ist,	Boolesch	Nein	`true`	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
	werden die Zielordner nicht ersetzt, wenn sich bereits ein Ordner am angegebenen Speicherort mit dem angegebenen Namen befindet.					

Eingabebeispiel: Kopieren eines Ordners (Linux)

```
name: CopyingAFolderLinux
action: CopyFolder
inputs:
  - source: /Sample/MyFolder/SampleFolder
    destination: /MyFolder/destinationFolder
```

Eingabebeispiel: Kopieren eines Ordners (Windows)

```
name: CopyingAFolderWindows
action: CopyFolder
inputs:
  - source: C:\Sample\MyFolder\SampleFolder
    destination: C:\MyFolder\destinationFolder
```

Eingabebeispiel: Kopieren eines Ordners mit dem Quellordnernamen (Linux)

```
name: CopyingFolderSourceFolderNameLinux
```

```
action: CopyFolder
inputs:
  - source: /Sample/MyFolder/SourceFolder
    destination: /MyFolder/
```

Eingabebeispiel: Kopieren eines Ordners unter Verwendung des Quellordnernamens (Windows)

```
name: CopyingFolderSourceFolderNameWindows
action: CopyFolder
inputs:
  - source: C:\Sample\MyFolder\SampleFolder
    destination: C:\MyFolder\
```

Eingabebeispiel: Kopieren eines Ordners mit dem Platzhalterzeichen (Linux)

```
name: CopyingFoldersWithWildCardLinux
action: CopyFolder
inputs:
  - source: /Sample/MyFolder/Sample*
    destination: /MyFolder/
```

Eingabebeispiel: Kopieren eines Ordners mit dem Platzhalterzeichen (Windows)

```
name: CopyingFoldersWithWildCardWindows
action: CopyFolder
inputs:
  - source: C:\Sample\MyFolder\Sample*
    destination: C:\MyFolder\
```

Eingabebeispiel: Kopieren eines Ordners ohne Überschreiben (Linux)

```
name: CopyingFoldersWithoutOverwriteLinux
action: CopyFolder
inputs:
  - source: /Sample/MyFolder/SourceFolder
    destination: /MyFolder/destinationFolder
    overwrite: false
```

Eingabebeispiel: Kopieren eines Ordners ohne Überschreiben (Windows)

```
name: CopyingFoldersWithoutOverwrite
```

```
action: CopyFolder
inputs:
  - source: C:\Sample\MyFolder\SourceFolder
    destination: C:\MyFolder\destinationFolder
    overwrite: false
```

Output

Keine.

CreateFile

Das CreateFile Aktionsmodul erstellt eine Datei an einem angegebenen Speicherort. Standardmäßig erstellt das Modul bei Bedarf auch rekursiv die übergeordneten Ordner.

Wenn die Datei bereits im angegebenen Ordner vorhanden ist, kürzt oder überschreibt das Aktionsmodul die vorhandene Datei standardmäßig. Sie können dieses Standardverhalten überschreiben, indem Sie die Überschreiboption auf setzen`false`. Wenn die Überschreiboption auf festgelegt ist `false`und bereits eine Datei am angegebenen Speicherort mit dem angegebenen Namen vorhanden ist, gibt das Aktionsmodul einen Fehler zurück.

Wenn sich der Wert der Dateikodierung vom Wert der Standardkodierung (`utf-8`) unterscheidet, können Sie den Wert der Dateikodierung mithilfe der `encoding` Option angeben. Standardmäßig `utf-32` wird davon ausgegangen, dass `utf-16` und die Little-Endian-Kodierung verwenden.

`owner`, `group`und `permissions` sind optionale Eingaben. Die Eingabe für `permissions` muss ein Zeichenfolgenwert sein. Dateien werden mit Standardwerten erstellt, wenn sie nicht bereitgestellt werden. Diese Optionen werden auf Windows-Plattformen nicht unterstützt. Dieses Aktionsmodul validiert und gibt einen Fehler zurück`owner`, wenn die `permissions` Optionen `group`, und auf Windows-Plattformen verwendet werden.

Dieses Aktionsmodul kann eine Datei mit Berechtigungen erstellen, die durch den `umask` Standardwert des Betriebssystems definiert sind. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, eine Datei oder einen Ordner im angegebenen übergeordneten Ordner zu erstellen.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja
`content`	Der Inhalt der Datei.	String	Nein	N/A	N/A	Ja
`encoding`	Der Codierungsstandard.	String	Nein	`utf8`	`utf8`, `utf-8`, `utf16utf-1` `utf16-LE`, `utf-16-LE` `utf16-BE`, `utf-16-BE` , `utf-32`, `utf32`, `utf32-LEutf-32-LE` , `utf32-BE`, und `utf-32-BE` . Bei dem Wert der Codierungsoption wird die Groß- und Kleinschreibung	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
					nicht beachtet.	
`owner`	Der Benutzername oder die ID.	String	Nein	N/A	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Gruppenname oder die ID.	String	Nein	Der aktuelle Benutzer.	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Dateiberechtigungen.	String	Nein	`0666`	N/A	Wird unter Windows nicht unterstützt.

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`overwrite`	Wenn der Name der angegebenen Datei bereits vorhanden ist, verhindert das Festlegen dieses Werts auf standardmäßig, `false` dass die Datei gekürzt oder überschrieben wird.	Boolesch	Nein	`true`	N/A	Ja

Eingabebeispiel: Erstellen einer Datei ohne Überschreiben (Linux)

```
name: CreatingFileWithoutOverwriteLinux
action: CreateFile
inputs:
  - path: /home/UserName/Sample.txt
    content: ABCD\nRandom\tvalues
    overwrite: false
```

Eingabebeispiel: Erstellen einer Datei ohne Überschreiben (Windows)

```
name: CreatingFileWithoutOverwriteWindows
action: CreateFile
inputs:
  - path: C:\Temp\Sample.txt
    content: ABCD\nRandom\tvalues
    overwrite: false
```

Eingabebeispiel: Erstellen einer Datei mit Dateieigenschaften

```
name: CreatingFileWithFileProperties
action: CreateFile
inputs:
  - path: SampleFolder/Sample.txt
    content: ABCD\nRandom\tvalues
    encoding: UTF-16
    owner: Ubuntu
    group: UbuntuGroup
    permissions: 0777
 - path: SampleFolder/SampleFile.txt
    permissions: 755
  - path: SampleFolder/TextFile.txt
    encoding: UTF-16
    owner: root
    group: rootUserGroup
```

Eingabebeispiel: Erstellen einer Datei ohne Dateieigenschaften

```
name: CreatingFileWithoutFileProperties
action: CreateFile
inputs:
  - path: ./Sample.txt
  - path: Sample1.txt
```

Eingabebeispiel: Erstellen Sie eine leere Datei, um einen Abschnitt im Linux-Cleanup-Skript zu überspringen

```
name: CreateSkipCleanupfile
action: CreateFile
inputs:
  - path: <skip section file name>
```

Weitere Informationen finden Sie unter Überschreiben des Linux-Bereinigungsskripts.

Output

Keine.

CreateFolder

Das CreateFolder Aktionsmodul erstellt einen Ordner an einem angegebenen Speicherort. Standardmäßig erstellt das Modul bei Bedarf auch rekursiv die übergeordneten Ordner.

Wenn der Ordner bereits im angegebenen Ordner vorhanden ist, kürzt oder überschreibt das Aktionsmodul standardmäßig den vorhandenen Ordner. Sie können dieses Standardverhalten überschreiben, indem Sie die Überschreiboption auf setzen`false`. Wenn die Überschreiboption auf festgelegt ist `false`und bereits ein Ordner am angegebenen Speicherort mit dem angegebenen Namen vorhanden ist, gibt das Aktionsmodul einen Fehler zurück.

`owner`, `group`und `permissions` sind optionale Eingaben. Die Eingabe für `permissions` muss ein Zeichenfolgenwert sein. Diese Optionen werden auf Windows-Plattformen nicht unterstützt. Dieses Aktionsmodul validiert und gibt einen Fehler zurück`owner`, wenn die `permissions` Optionen `group`, und auf Windows-Plattformen verwendet werden.

Dieses Aktionsmodul kann einen Ordner mit Berechtigungen erstellen, die durch den `umask` Standardwert des Betriebssystems definiert sind. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, einen Ordner am angegebenen Speicherort zu erstellen.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`owner`	Der Benutzername oder die ID.	String	Nein	Der aktuelle Benutzer.	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Gruppenname oder die ID.	String	Nein	Die Gruppe des aktuellen Benutzers.	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Ordnerberechtigungen.	String	Nein	`0777`	N/A	Wird unter Windows nicht unterstützt.

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`overwrite`	Wenn der Name der angegebenen Datei bereits vorhanden ist, verhindert das Festlegen dieses Werts auf standardmäßig, `false` dass die Datei gekürzt oder überschrieben wird.	Boolesch	Nein	`true`	N/A	Ja

Eingabebeispiel: Erstellen eines Ordners (Linux)

```
name: CreatingFolderLinux
action: CreateFolder
inputs:
  - path: /Sample/MyFolder/
```

Eingabebeispiel: Erstellen eines Ordners (Windows)

```
name: CreatingFolderWindows
action: CreateFolder
```

```
inputs:
  - path: C:\MyFolder
```

Eingabebeispiel: Erstellen eines Ordners mit Ordnereigenschaften

```
name: CreatingFolderWithFolderProperties
action: CreateFolder
inputs:
  - path: /Sample/MyFolder/Sample/
    owner: SampleOwnerName
    group: SampleGroupName
    permissions: 0777
  - path: /Sample/MyFolder/SampleFoler/
    permissions: 777
```

Eingabebeispiel: Erstellen Sie einen Ordner, der den vorhandenen Ordner überschreibt, falls vorhanden.

```
name: CreatingFolderWithOverwrite
action: CreateFolder
inputs:
  - path: /Sample/MyFolder/Sample/
    overwrite: true
```

Output

Keine.

CreateSymlink

Das CreateSymlink Aktionsmodul erstellt symbolische Links oder Dateien, die einen Verweis auf eine andere Datei enthalten. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Die Eingabe für die `target` Optionen `path` und kann entweder ein absoluter oder relativer Pfad sein. Wenn es sich bei der Eingabe für die `path` Option um einen relativen Pfad handelt, wird sie beim Erstellen des Links durch den absoluten Pfad ersetzt.

Wenn im angegebenen Ordner bereits ein Link mit dem angegebenen Namen vorhanden ist, gibt das Aktionsmodul standardmäßig einen Fehler zurück. Sie können dieses Standardverhalten überschreiben, indem Sie die `force` Option auf setzen`true`. Wenn die `force` Option auf festgelegt ist`true`, überschreibt das Modul den vorhandenen Link.

Wenn kein übergeordneter Ordner vorhanden ist, erstellt das Aktionsmodul den Ordner standardmäßig rekursiv.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Die Zieldatei ist zur Laufzeit nicht vorhanden.
- Eine nichtsymbolische Linkdatei mit dem angegebenen Namen ist bereits vorhanden.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`target`	Der Zieldateipfad, auf den der symbolische Link verweist.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`force`	Erzwingt die Erstellung eines Links, wenn bereits ein Link mit demselben Namen	Boolesch	Nein	`false`	N/A	Wird unter Windows nicht unterstützt.

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
	vorhanden ist.					

Eingabebeispiel: Erstellen eines symbolischen Links, der die Erstellung eines Links erzwingt

```
name: CreatingSymbolicLinkWithForce
action: CreateSymlink
inputs:
  - path: /Folder2/Symboliclink.txt
    target: /Folder/Sample.txt
    force: true
```

Eingabebeispiel: Erstellen Sie einen symbolischen Link, der die Erstellung eines Links nicht erzwingt

```
name: CreatingSymbolicLinkWithOutForce
action: CreateSymlink
inputs:
  - path: Symboliclink.txt
    target: /Folder/Sample.txt
```

Output

Keine.

DeleteFile

Das DeleteFile Aktionsmodul löscht eine oder mehrere Dateien an einem angegebenen Speicherort.

Die Eingabe von `path` sollte ein gültiger Dateipfad oder ein Dateipfad mit einem Platzhalterzeichen (*) im Dateinamen sein. Wenn Platzhalterzeichen im Dateinamen angegeben werden, werden alle Dateien innerhalb desselben Ordners, die mit dem Platzhalter übereinstimmen, gelöscht.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, Löschvorgänge durchzuführen.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja

Eingabebeispiel: Löschen einer einzelnen Datei (Linux)

```
name: DeletingSingleFileLinux
action: DeleteFile
inputs:
  - path: /SampleFolder/MyFolder/Sample.txt
```

Eingabebeispiel: Löschen einer einzelnen Datei (Windows)

```
name: DeletingSingleFileWindows
action: DeleteFile
inputs:
  - path: C:\SampleFolder\MyFolder\Sample.txt
```

Eingabebeispiel: Löschen einer Datei, die mit „log“ endet (Linux)

```
name: DeletingFileEndingWithLogLinux
action: DeleteFile
inputs:
  - path: /SampleFolder/MyFolder/*log
```

Eingabebeispiel: Löschen einer Datei, die mit „log“ endet (Windows)

```
name: DeletingFileEndingWithLogWindows
action: DeleteFile
inputs:
  - path: C:\SampleFolder\MyFolder\*log
```

Eingabebeispiel: Löschen aller Dateien in einem angegebenen Ordner (Linux)

```
name: DeletingAllFilesInAFolderLinux
```

```
action: DeleteFile
inputs:
  - path: /SampleFolder/MyFolder/*
```

Eingabebeispiel: Löschen aller Dateien in einem angegebenen Ordner (Windows)

```
name: DeletingAllFilesInAFolderWindows
action: DeleteFile
inputs:
  - path: C:\SampleFolder\MyFolder\*
```

Output

Keine.

DeleteFolder

Das DeleteFolder Aktionsmodul löscht Ordner.

Wenn der Ordner nicht leer ist, müssen Sie die `force` Option auf setzen, `true` um den Ordner und seinen Inhalt zu entfernen. Wenn Sie die `force` Option nicht auf festlegen und der Ordner`true`, den Sie löschen möchten, nicht leer ist, gibt das Aktionsmodul einen Fehler zurück. Der Standardwert der `force` Option ist `false`.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, Löschvorgänge durchzuführen.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Ja
`force`	Entfernt den	Boolesch	Nein	`false`	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
	Ordner, unabhängig davon, ob der Ordner leer ist oder nicht.					

Eingabebeispiel: Löschen eines Ordners, der nicht leer ist, mit der **`force`** Option (Linux)

```
name: DeletingFolderWithForceOptionLinux
action: DeleteFolder
inputs:
  - path: /Sample/MyFolder/Sample/
    force: true
```

Eingabebeispiel: Löschen eines Ordners, der nicht leer ist, mit der **`force`** Option (Windows)

```
name: DeletingFolderWithForceOptionWindows
action: DeleteFolder
inputs:
  - path: C:\Sample\MyFolder\Sample\
    force: true
```

Eingabebeispiel: Löschen eines Ordners (Linux)

```
name: DeletingFolderWithOutForceLinux
action: DeleteFolder
inputs:
  - path: /Sample/MyFolder/Sample/
```

Eingabebeispiel: Löschen eines Ordners (Windows)

```
name: DeletingFolderWithOutForce
action: DeleteFolder
```

```
inputs:
  - path: C:\Sample\MyFolder\Sample\
```

Output

Keine.

ListFiles

Das ListFiles Aktionsmodul listet die Dateien in einem angegebenen Ordner auf. Wenn die rekursive Option auf festgelegt ist`true`, werden die Dateien in Unterordnern aufgelistet. In diesem Modul werden Dateien in Unterordnern standardmäßig nicht aufgelistet.

Um alle Dateien mit Namen aufzulisten, die einem bestimmten Muster entsprechen, verwenden Sie die `fileNamePattern` Option , um das Muster bereitzustellen. Die `fileNamePattern` Option akzeptiert den Platzhalterwert (*). Wenn bereitgestellt `fileNamePattern` wird, werden alle Dateien zurückgegeben, die dem angegebenen Dateinamenformat entsprechen.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Der angegebene Ordner ist zur Laufzeit nicht vorhanden.
- Sie sind nicht berechtigt, eine Datei oder einen Ordner im angegebenen übergeordneten Ordner zu erstellen.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Ja
`fileNamePattern`	Das abzugleichende Muster,	String	Nein	N/A	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
	um alle Dateien aufzulisten, deren Namen dem Muster entsprechen.					
`recursive`	Listet Dateien im Ordner rekursiv auf.	Boolesch	Nein	`false`	N/A	Ja

Eingabebeispiel: Auflisten von Dateien im angegebenen Ordner (Linux)

```
name: ListingFilesInSampleFolderLinux
action: ListFiles
inputs:
  - path: /Sample/MyFolder/Sample
```

Eingabebeispiel: Auflisten von Dateien im angegebenen Ordner (Windows)

```
name: ListingFilesInSampleFolderWindows
action: ListFiles
inputs:
  - path: C:\Sample\MyFolder\Sample
```

Eingabebeispiel: Auflisten von Dateien, die mit „log“ enden (Linux)

```
name: ListingFilesWithEndingWithLogLinux
action: ListFiles
```

```
inputs:
  - path: /Sample/MyFolder/
    fileNamePattern: *log
```

Eingabebeispiel: Auflisten von Dateien, die mit „log“ enden (Windows)

```
name: ListingFilesWithEndingWithLogWindows
action: ListFiles
inputs:
  - path: C:\Sample\MyFolder\
    fileNamePattern: *log
```

Eingabebeispiel: Dateien rekursiv auflisten

```
name: ListingFilesRecursively
action: ListFiles
inputs:
  - path: /Sample/MyFolder/
    recursive: true
```

Output

Primitiv	Beschreibung	Typ
`files`	Die Liste der Dateien.	String

Output example

```
{
  "files": "/sample1.txt,/sample2.txt,/sample3.txt"
}
```

MoveFile

Das MoveFile Aktionsmodul verschiebt Dateien von der angegebenen Quelle in das angegebene Ziel.

Wenn die Datei bereits im angegebenen Ordner vorhanden ist, überschreibt das Aktionsmodul standardmäßig die vorhandene Datei. Sie können dieses Standardverhalten überschreiben, indem Sie die Überschreiboption auf setzen`false`. Wenn die Überschreiboption auf festgelegt ist `false`und bereits eine Datei am angegebenen Speicherort mit dem angegebenen Namen vorhanden ist, gibt das Aktionsmodul einen Fehler zurück. Diese Option funktioniert genauso wie der `mv` Befehl unter Linux, der standardmäßig überschreibt.

Der Name der Quelldatei kann einen Platzhalter (*) enthalten. Platzhalterzeichen werden erst nach dem letzten Dateipfadtrennzeichen (/ oder \) akzeptiert. Wenn Platzhalterzeichen im Quelldateinamen enthalten sind, werden alle Dateien, die mit dem Platzhalter übereinstimmen, in den Zielordner kopiert. Wenn Sie mehr als eine Datei mithilfe eines Platzhalterzeichens verschieben möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/ oder \) enden, das angibt, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Zieldateiname vom Quelldateinamen unterscheidet, können Sie den Zieldateinamen mit der `destination` Option angeben. Wenn Sie keinen Zieldateinamen angeben, wird der Name der Quelldatei verwendet, um die Zieldatei zu erstellen. Jeder Text, der dem letzten Dateipfadtrennzeichen (/ oder \) folgt, wird als Dateiname behandelt. Wenn Sie denselben Dateinamen wie die Quelldatei verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/ oder \) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, eine Datei im angegebenen Ordner zu erstellen.
- Die Quelldateien sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Dateinamen und die `overwrite` Option ist auf gesetzt`false`.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`source`	Der Pfad der Quelldatei.	String	Ja	–	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`destination`	Der Pfad der Zieldatei.	String	Ja	–	N/A	Ja
`overwrite`	Wenn der Wert auf „false“ gesetzt ist, werden die Zieldateien nicht ersetzt, wenn sich bereits eine Datei am angegebenen Speicherort mit dem angegebenen Namen befindet.	Boolesch	Nein	`true`	N/A	Ja

Eingabebeispiel: Verschieben einer Datei (Linux)

```
name: MovingAFileLinux
action: MoveFile
inputs:
  - source: /Sample/MyFolder/Sample.txt
    destination: /MyFolder/destinationFile.txt
```

Eingabebeispiel: Verschieben einer Datei (Windows)

```
name: MovingAFileWindows
action: MoveFile
inputs:
  - source: C:\Sample\MyFolder\Sample.txt
    destination: C:\MyFolder\destinationFile.txt
```

Eingabebeispiel: Verschieben einer Datei mit dem Quelldateinamen (Linux)

```
name: MovingFileWithSourceFileNameLinux
action: MoveFile
inputs:
  - source: /Sample/MyFolder/Sample.txt
    destination: /MyFolder/
```

Eingabebeispiel: Verschieben einer Datei mithilfe des Quelldateinamens (Windows)

```
name: MovingFileWithSourceFileNameWindows
action: MoveFile
inputs:
  - source: C:\Sample\MyFolder\Sample.txt
    destination: C:\MyFolder
```

Eingabebeispiel: Verschieben einer Datei mit einem Platzhalterzeichen (Linux)

```
name: MovingFilesWithWildCardLinux
action: MoveFile
inputs:
  - source: /Sample/MyFolder/Sample*
    destination: /MyFolder/
```

Eingabebeispiel: Verschieben einer Datei mit einem Platzhalterzeichen (Windows)

```
name: MovingFilesWithWildCardWindows
action: MoveFile
inputs:
  - source: C:\Sample\MyFolder\Sample*
    destination: C:\MyFolder
```

Eingabebeispiel: Verschieben einer Datei ohne Überschreiben (Linux)

```
name: MovingFilesWithoutOverwriteLinux
```

```
action: MoveFile
inputs:
  - source: /Sample/MyFolder/Sample.txt
    destination: /MyFolder/destinationFile.txt
    overwrite: false
```

Eingabebeispiel: Verschieben einer Datei ohne Überschreiben (Windows)

```
name: MovingFilesWithoutOverwrite
action: MoveFile
inputs:
  - source: C:\Sample\MyFolder\Sample.txt
    destination: C:\MyFolder\destinationFile.txt
    overwrite: false
```

Output

Keine.

MoveFolder

Das MoveFolder Aktionsmodul verschiebt Ordner von der angegebenen Quelle zum angegebenen Ziel. Die Eingabe für die `source` Option ist der zu verschiebende Ordner, und die Eingabe für die `destination` Option ist der Ordner, in den der Inhalt der Quellordner verschoben wird.

Wenn der übergeordnete Zielordner oder die Eingabe für die `destination` Option zur Laufzeit nicht vorhanden ist, besteht das Standardverhalten des Moduls darin, den Ordner rekursiv am angegebenen Ziel zu erstellen.

Wenn im Zielordner bereits ein Ordner mit demselben wie der Quellordner vorhanden ist, überschreibt das Aktionsmodul standardmäßig den vorhandenen Ordner. Sie können dieses Standardverhalten überschreiben, indem Sie die Überschreiboption auf setzen`false`. Wenn die Überschreiboption auf festgelegt ist `false`und bereits ein Ordner am angegebenen Speicherort mit dem angegebenen Namen vorhanden ist, gibt das Aktionsmodul einen Fehler zurück.

Der Name des Quellordners kann einen Platzhalter (`*`) enthalten. Platzhalterzeichen werden erst nach dem letzten Dateipfadtrennzeichen (`/` oder `\`) akzeptiert. Wenn Platzhalterzeichen im Quellordnernamen enthalten sind, werden alle Ordner, die mit dem Platzhalter übereinstimmen, in den Zielordner kopiert. Wenn Sie mehr als einen Ordner mithilfe eines Platzhalterzeichens verschieben möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (`/` oder `\`) enden, das angibt, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Name des Zielordners vom Namen des Quellordners unterscheidet, können Sie den Namen des Zielordners mit der `destination` Option angeben. Wenn Sie keinen Zielordnernamen angeben, wird der Name des Quellordners verwendet, um den Zielordner zu erstellen. Jeder Text, der dem letzten Dateipfadtrennzeichen (/ oder \) folgt, wird als Ordnername behandelt. Wenn Sie denselben Ordnernamen wie der Quellordner verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/ oder \) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, einen Ordner im Zielordner zu erstellen.
- Die Quellordner sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Namen und die `overwrite` Option ist auf gesetzt`false`.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`source`	Der Quellordnerpfad.	String	Ja	–	N/A	Ja
`destination`	Der Pfad des Zielordners.	String	Ja	–	N/A	Ja
`overwrite`	Wenn der Wert auf „false“ gesetzt ist, werden die Zielordner nicht ersetzt,	Boolesch	Nein	`true`	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
	wenn sich bereits ein Ordner am angegebenen Speicherort mit dem angegebenen Namen befindet.					

Eingabebeispiel: Verschieben eines Ordners (Linux)

```
name: MovingAFolderLinux
action: MoveFolder
inputs:
  - source: /Sample/MyFolder/SourceFolder
    destination: /MyFolder/destinationFolder
```

Eingabebeispiel: Verschieben eines Ordners (Windows)

```
name: MovingAFolderWindows
action: MoveFolder
inputs:
  - source: C:\Sample\MyFolder\SourceFolder
    destination: C:\MyFolder\destinationFolder
```

Eingabebeispiel: Verschieben eines Ordners unter Verwendung des Quellordnernamens (Linux)

```
name: MovingFolderWithSourceFolderNameLinux
action: MoveFolder
inputs:
  - source: /Sample/MyFolder/SampleFolder
    destination: /MyFolder/
```

Eingabebeispiel: Verschieben eines Ordners unter Verwendung des Quellordnernamens (Windows)

```
name: MovingFolderWithSourceFolderNameWindows
action: MoveFolder
inputs:
  - source: C:\Sample\MyFolder\SampleFolder
    destination: C:\MyFolder\
```

Eingabebeispiel: Verschieben eines Ordners mit einem Platzhalterzeichen (Linux)

```
name: MovingFoldersWithWildCardLinux
action: MoveFolder
inputs:
  - source: /Sample/MyFolder/Sample*
    destination: /MyFolder/
```

Eingabebeispiel: Verschieben eines Ordners mit einem Platzhalterzeichen (Windows)

```
name: MovingFoldersWithWildCardWindows
action: MoveFolder
inputs:
  - source: C:\Sample\MyFolder\Sample*
    destination: C:\MyFolder\
```

Eingabebeispiel: Verschieben eines Ordners ohne Überschreiben (Linux)

```
name: MovingFoldersWithoutOverwriteLinux
action: MoveFolder
inputs:
  - source: /Sample/MyFolder/SampleFolder
    destination: /MyFolder/destinationFolder
    overwrite: false
```

Eingabebeispiel: Verschieben eines Ordners ohne Überschreiben (Windows)

```
name: MovingFoldersWithoutOverwriteWindows
action: MoveFolder
inputs:
  - source: C:\Sample\MyFolder\SampleFolder
    destination: C:\MyFolder\destinationFolder
    overwrite: false
```

Output

Keine.

ReadFile

Das ReadFile Aktionsmodul liest den Inhalt einer Textdatei vom Typ Zeichenfolge. Dieses Modul kann verwendet werden, um den Inhalt einer Datei zur Verwendung in nachfolgenden Schritten durch Verketten oder zum Lesen von Daten in der `console.log` Datei zu lesen. Wenn der angegebene Pfad ein symbolischer Link ist, gibt dieses Modul den Inhalt der Zieldatei zurück. Dieses Modul unterstützt nur Textdateien.

Wenn sich der Wert der Dateikodierung vom Wert der Standardkodierung (`utf-8`) unterscheidet, können Sie den Wert der Dateikodierung mithilfe der `encoding` Option angeben. Standardmäßig `utf-32` wird davon ausgegangen, dass `utf-16` und die Little-Endian-Kodierung verwenden.

Standardmäßig kann dieses Modul den Dateiinhalt nicht in die `console.log` Datei drucken. Sie können diese Einstellung überschreiben, indem Sie die -`printFileContent`Eigenschaft auf setzen`true`.

Dieses Modul kann nur den Inhalt einer Datei zurückgeben. Es kann keine Dateien wie Excel- oder JSON-Dateien analysieren.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Die Datei ist zur Laufzeit nicht vorhanden.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja
`encoding`	Der Codierungsstandard.	String	Nein	`utf8`	`utf8`, `utf-8`, `utf16utf-1`	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
					`utf16-LE`, `utf-16-LE` `utf16-BE`, `utf-16-BE` , `utf-32`, `utf32`, `utf32-LEutf-32-LE` , `utf32-BE`, und `utf-32-BE` . Bei dem Wert der Codierungsoption wird die Groß- und Kleinschreibung nicht beachtet.	
`printFileContent`	Druckt den Dateiinhalt in die `console.log` Datei.	Boolesch	Nein	false	N/A	Ja.

Eingabebeispiel: Lesen einer Datei (Linux)

```
name: ReadingFileLinux
action: ReadFile
inputs:
  - path: /home/UserName/SampleFile.txt
```

Eingabebeispiel: Lesen einer Datei (Windows)

```
name: ReadingFileWindows
action: ReadFile
inputs:
  - path: C:\Windows\WindowsUpdate.log
```

Eingabebeispiel: Lesen einer Datei und Angeben des Kodierungsstandards

```
name: ReadingFileWithFileEncoding
action: ReadFile
inputs:
  - path: /FolderName/SampleFile.txt
    encoding: UTF-32
```

Eingabebeispiel: Lesen einer Datei und Drucken in der **`console.log`** Datei

```
name: ReadingFileToConsole
action: ReadFile
inputs:
  - path: /home/UserName/SampleFile.txt
    printFileContent: true
```

Output

Feld	Beschreibung	Typ
`content`	Der Inhalt der Datei.	Zeichenfolge

Output example

```
{
```

```
    "content" : "The file content"
}
```

SetFileEncoding

Das SetFileEncoding Aktionsmodul ändert die Codierungseigenschaft einer vorhandenen Datei. Dieses Modul kann die Dateikodierung von `utf-8` in einen angegebenen Kodierungsstandard konvertieren. Standardmäßig `utf-32` wird davon ausgegangen, dass `utf-16` und eine Little-Endian-Kodierung sind.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Die Datei ist zur Laufzeit nicht vorhanden.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja
`encoding`	Der Codierungsstandard.	String	Nein	`utf8`	`utf8`, `utf-8`, `utf16utf-1` `utf16-LE`, `utf-16-LE` `utf16-BE`, `utf-16-BE` , `utf-32`, `utf32`, `utf32-LEutf-32-`	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
					`LE` , `utf32-BE`, und `utf-32-BE` . Bei dem Wert der Codierungsoption wird die Groß- und Kleinschreibung nicht beachtet.	

Eingabebeispiel: Festlegen der Dateikodierungseigenschaft

```
name: SettingFileEncodingProperty
action: SetFileEncoding
inputs:
  - path: /home/UserName/SampleFile.txt
    encoding: UTF-16
```

Output

Keine.

SetFileOwner

Das SetFileOwner Aktionsmodul ändert die Eigenschaften `owner` und`group` des Eigentümers einer vorhandenen Datei. Wenn die angegebene Datei ein symbolischer Link ist, ändert das Modul die -`owner`Eigenschaft der Quelldatei. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Dieses Modul akzeptiert Benutzer- und Gruppennamen als Eingaben. Wenn der Gruppenname nicht angegeben wird, weist das Modul den Gruppenbesitzer der Datei der Gruppe zu, zu der der Benutzer gehört.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Der angegebene Benutzer- oder Gruppenname ist zur Laufzeit nicht vorhanden.
- Die Datei ist zur Laufzeit nicht vorhanden.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`owner`	Der Benutzername	Zeichenfolge	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Name der Gruppe.	String	Nein	Der Name der Gruppe, zu der der Benutzer gehört.	N/A	Wird unter Windows nicht unterstützt.

Eingabebeispiel: Legen Sie die Eigenschaft des Dateieigentümers fest, ohne den Namen der Gruppe anzugeben

```
name: SettingFileOwnerPropertyNoGroup
action: SetFileOwner
inputs:
  - path: /home/UserName/SampleText.txt
    owner: LinuxUser
```

Eingabebeispiel: Legen Sie die Eigenschaft des Dateieigentümers fest, indem Sie den Eigentümer und die Benutzergruppe angeben

```
name: SettingFileOwnerProperty
action: SetFileOwner
inputs:
  - path: /home/UserName/SampleText.txt
    owner: LinuxUser
    group: LinuxUserGroup
```

Output

Keine.

SetFolderOwner

Das SetFolderOwner Aktionsmodul ändert rekursiv die Eigenschaften `owner` und`group` des Eigentümers eines vorhandenen Ordners. Standardmäßig kann das Modul die Eigentümerschaft für alle Inhalte in einem Ordner ändern. Sie können die `recursive` Option auf setzen`false`, um dieses Verhalten zu überschreiben. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Dieses Modul akzeptiert Benutzer- und Gruppennamen als Eingaben. Wenn der Gruppenname nicht angegeben wird, weist das Modul den Gruppenbesitzer der Datei der Gruppe zu, zu der der Benutzer gehört.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Der angegebene Benutzer- oder Gruppenname ist zur Laufzeit nicht vorhanden.
- Der Ordner ist zur Laufzeit nicht vorhanden.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`owner`	Der Benutzername	Zeichenfolge	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Name der Gruppe.	String	Nein	Der Name der Gruppe, zu der der Benutzer gehört.	N/A	Wird unter Windows nicht unterstützt.
`recursive`	Überschreibt das Standardverhalten, bei dem der Besitz für den gesamten Inhalt eines Ordners geändert wird, wenn er auf festgelegt ist`false`.	Boolesch	Nein	`true`	N/A	Wird unter Windows nicht unterstützt.

Eingabebeispiel: Legen Sie die Eigenschaft des Ordnereigentümers fest, ohne den Namen der Gruppe anzugeben

```
name: SettingFolderPropertyWithOutGroup
action: SetFolderOwner
inputs:
  - path: /SampleFolder/
    owner: LinuxUser
```

Eingabebeispiel: Legen Sie die Eigenschaft des Ordnereigentümers fest, ohne die Eigentümerschaft aller Inhalte in einem Ordner zu überschreiben

```
name: SettingFolderPropertyWithOutRecursively
action: SetFolderOwner
inputs:
  - path: /SampleFolder/
    owner: LinuxUser
    recursive: false
```

Eingabebeispiel: Legen Sie die Eigenschaft für den Dateibesitz fest, indem Sie den Namen der Gruppe angeben

```
name: SettingFolderPropertyWithGroup
action: SetFolderOwner
inputs:
  - path: /SampleFolder/
    owner: LinuxUser
    group: LinuxUserGroup
```

Output

Keine.

SetFilePermissions

Das SetFilePermissions Aktionsmodul ändert die `permissions` einer vorhandenen Datei. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Die Eingabe für `permissions` muss ein Zeichenfolgenwert sein.

Dieses Aktionsmodul kann eine Datei mit Berechtigungen erstellen, die durch den standardmäßigen Umask-Wert des Betriebssystems definiert sind. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Die Datei ist zur Laufzeit nicht vorhanden.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Dateipfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Dateiberechtigungen.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.

Eingabebeispiel: Ändern von Dateiberechtigungen

```
name: ModifyingFilePermissions
action: SetFilePermissions
inputs:
  - path: /home/UserName/SampleFile.txt
    permissions: 766
```

Output

Keine.

SetFolderPermissions

Das SetFolderPermissions Aktionsmodul ändert rekursiv die `permissions` eines vorhandenen Ordners und aller Unterdateien und Unterordner. Standardmäßig kann dieses Modul Berechtigungen für den gesamten Inhalt des angegebenen Ordners ändern. Sie können die `recursive` Option auf setzen`false`, um dieses Verhalten zu überschreiben. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Die Eingabe für `permissions` muss ein Zeichenfolgenwert sein.

Dieses Aktionsmodul kann Berechtigungen entsprechend dem Standard-Umask-Wert des Betriebssystems ändern. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes auftritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Der Ordner ist zur Laufzeit nicht vorhanden.
- Beim Ausführen des Vorgangs stößt das Aktionsmodul auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Ordnerberechtigungen.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`recursive`	Überschreibt das Standardv	Boolesch	Nein	`true`	N/A	Wird unter Windows

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Unterstützt auf allen Plattformen
	erhalten beim Ändern von Berechtigungen für den gesamten Inhalt eines Ordners, wenn dieser auf festgelegt ist`false`.					nicht unterstützt.

Eingabebeispiel: Festlegen von Ordnerberechtigungen

```
name: SettingFolderPermissions
action: SetFolderPermissions
inputs:
  - path: SampleFolder/
    permissions: 0777
```

Eingabebeispiel: Festlegen von Ordnerberechtigungen, ohne die Berechtigungen für den gesamten Inhalt eines Ordners zu ändern

```
name: SettingFolderPermissionsNoRecursive
action: SetFolderPermissions
inputs:
  - path: /home/UserName/SampleFolder/
    permissions: 777
    recursive: false
```

Output

Keine.

Softwareinstallationsaktionen

In diesem Abschnitt werden Aktionsmodule beschrieben, die Befehle und Anweisungen für Softwareinstallationsaktionen ausführen.

IAM-Anforderungen

Wenn es sich bei Ihrem Installations-Download-Pfad um einen S3-URI handelt, muss die IAM-Rolle, die Sie Ihrem Instance-Profil zuordnen, über die Berechtigung verfügen, das `S3Download` Aktionsmodul auszuführen. Um die erforderliche Berechtigung zu erteilen, fügen Sie die `S3:GetObject` IAM-Richtlinie der IAM-Rolle an, die Ihrem Instance-Profil zugeordnet ist, und geben Sie den Pfad für Ihren Bucket an. Beispiel, *`arn:aws:s3:::BucketName/*`*).

Komplexe MSI-Eingaben

Wenn Ihre Eingabezeichenfolgen doppelte Anführungszeichen ('') enthalten, müssen Sie eine der folgenden Methoden verwenden, um sicherzustellen, dass sie korrekt interpretiert werden:

- Sie können einfache Anführungszeichen (') außerhalb Ihrer Zeichenfolge verwenden, um sie einzuschließen, und doppelte Anführungszeichen (") innerhalb Ihrer Zeichenfolge verwenden, wie im folgenden Beispiel gezeigt.

  ```
  properties:
    COMPANYNAME: '"Acme ""Widgets"" and ""Gizmos."""'
  ```

  Wenn Sie in diesem Fall ein Apostroph innerhalb Ihrer Zeichenfolge verwenden müssen, müssen Sie es mit einem Escape-Zeichen versehen. Das bedeutet, dass ein anderes einfaches Anführungszeichen (') vor dem Apostroph verwendet wird.
- Sie können doppelte Anführungszeichen (") außerhalb Ihrer Zeichenfolge verwenden, um sie einzuschließen. Und Sie können alle doppelten Anführungszeichen innerhalb Ihrer Zeichenfolge mit dem umgekehrten Schrägstrich (\) maskieren, wie im folgenden Beispiel gezeigt.

  ```
  properties:
    COMPANYNAME: "\"Acme \"\"Widgets\"\" and \"\"Gizmos.\"\"\""
  ```

Beide Methoden übergeben den Wert `COMPANYNAME="Acme ""Widgets"" and ""Gizmos."""` an den msiexec Befehl .

Softwareinstallationsaktionsmodule

- InstallMSI
- UninstallMSI

InstallMSI

Das `InstallMSI` Aktionsmodul installiert eine Windows-Anwendung mithilfe einer MSI-Datei. Sie können die MSI-Datei mit einem lokalen Pfad, einem S3-Objekt-URI oder einer Web-URL angeben. Die Neustartoption konfiguriert das Neustartverhalten des Systems.

AWSTOE generiert den msiexec Befehl basierend auf den Eingabeparametern für das Aktionsmodul. Werte für die Eingabeparameter `path` (MSI-Dateispeicherort) und `logFile` (Protokolldateispeicherort) müssen in Anführungszeichen (") eingeschlossen werden.

Die folgenden MSI-Beendigungscodes gelten als erfolgreich:

- 0 (Erfolg)
- 1614 (ERROR_PRODUCT_UNINSTALLED)
- 1641 (Neustart initiiert)
- 3010 (Neustart erforderlich)

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`path`	Geben Sie den Speicherort der MSI-Datei mit einer der folgenden Optionen an: • Der lokale Dateipfad. Der Pfad	String	Ja	–	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	kann absolute oder relative sein. • Ein gültiger S3-Objekt-URI. • Eine gültige Web-HTTP/HTTPS-URL (HTTPS wird empfohlen), die dem RFC-3986-Standard entspricht. Verkettungsausdrücke sind zulässig.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`reboot`	Konfigurieren Sie das Systemneustartverhalten, das auf eine erfolgreiche Ausführung des Aktionsmoduls folgt. Einstellungen: • `Force` – Initiiert einen Systemneustart, nachdem der msiexec Befehl erfolgreich ausgeführt wurde. • `Allow` – Startet einen Systemneustart,	String	Nein	`Allow`	`Allow`, `Force`, `Skip`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	wenn der msiexec Befehl einen Beendigungscode zurückgibt, der angibt, dass ein Neustart erforderlich ist. • `Skip` – Protokolliert eine Informationsmeldung in der `-console.log` Datei, die angibt, dass ein Neustart übersprungen wurde. Diese Option verhindert einen Neustart, auch wenn der msiexec				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Befehl einen Beendigungscode zurückgibt, der angibt, dass ein Neustart erforderlich ist.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logOptions`	Geben Sie die Optionen an, die für die MSI-Installationsprotokollierung verwendet werden sollen. Angegebene Flags werden zusammen mit dem `/L` Befehlszeilenparameter an das MSI-Installationsprogramm übergeben, um die Protokollierung zu aktivieren. Wenn keine Flags angegeben sind, AWSTOE verwendet den Standardwert.	String	Nein	`*VX`	`i,w,e,a,r,u,c,m,o,p,v,x,+,!,*`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Weitere Informationen zu Protokoll optionen für MSI finden Sie unter Befehlsze ilenoptio nen in der Produktdo kumentati on zum Microsoft Windows Installer.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logFile`	Ein absoluter oder relativer Pfad zum Speicherort der Protokoll datei. Wenn der Protokoll dateipfad nicht vorhanden ist, wird er erstellt. Wenn der Protokoll dateipfad nicht angegeben wird, speichert das MSI-Installationsp rotokoll AWSTOE nicht.	String	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`properties`	Schlüssel-Wert-Paare der MSI-Protokollierungseigenschaft, zum Beispiel: `TARGETDIR: "C:\target\location"` Hinweis: Die Änderung der folgenden Eigenschaften ist nicht zulässig: • `REBOOT="ReallySupress"` • `REINSTALLMODE="ecmus"` • `REINSTALL="ALL"`	Map[Zeichenfolge]Zeichenfolge	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`ignoreAuthenticodeSignatureErrors`	Flag, um Fehler bei der Validierung der Authentifizierungssignatur für das im Pfad angegebene Installationsprogramm zu ignorieren. Der Get-AuthenticodeSignature Befehl wird verwendet, um die Installationsprogramme zu validieren. Einstellungen: • `true` – Validierungsfehler werden ignoriert und das	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Installationsprogramm wird ausgeführt. • `false` – Validierungsfehler werden nicht ignoriert. Das Installationsprogramm wird nur ausgeführt, wenn die Validierung erfolgreich ist. Dies ist das Standardverhalten.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`allowUnsignedInstaller`	Flag, um die Ausführung des im Pfad angegebenen unsignierten Installationsprogramms zu ermöglichen. Der Get-AuthenticodeSignature Befehl wird verwendet, um die Installationsprogramme zu validieren. Einstellungen: • `true` – Ignoriert den vom Get-AuthenticodeSignature Befehl zurückgegebenen	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	`NotSigned` Status und führt das Installationsprogramm aus. • `false` – erfordert, dass das Installationsprogramm signiert ist. Unsignierte Installationsprogramme werden nicht ausgeführt. Dies ist das Standardverhalten.				

Beispiele

Die folgenden Beispiele zeigen je nach Installationspfad Variationen des Eingabeabschnitts für Ihr Komponentendokument.

Eingabebeispiel: Installation des lokalen Dokumentpfads

```
- name: local-path-install
  steps:
```

```
    - name: LocalPathInstaller
      action: InstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-install.log
        logOptions: '*VX'
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

Eingabebeispiel: Installation des Amazon S3-Pfads

```
- name: s3-path-install
  steps:
    - name: S3PathInstaller
      action: InstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-install.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

Eingabebeispiel: Installation des Webpfads

```
- name: web-path-install
  steps:
    - name: WebPathInstaller
      action: InstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-install.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: false
```

Output

Im Folgenden finden Sie ein Beispiel für die Ausgabe des -`InstallMSI`Aktionsmoduls.

```
{
```

```
  "logFile": "web-path-install.log",
  "msiExitCode": 0,
  "stdout": ""
}
```

UninstallMSI

Mit dem `UninstallMSI` Aktionsmodul können Sie eine Windows-Anwendung mithilfe einer MSI-Datei entfernen. Sie können den Speicherort der MSI-Datei mithilfe eines lokalen Dateipfads, eines S3-Objekt-URI oder einer Web-URL angeben. Die Neustartoption konfiguriert das Neustartverhalten des Systems.

AWSTOE generiert den msiexec Befehl basierend auf den Eingabeparametern für das Aktionsmodul. Der Speicherort der MSI-Datei (`path`) und der Speicherort der Protokolldatei (`logFile`) werden explizit in doppelte Anführungszeichen (") eingeschlossen, während der msiexec Befehl generiert wird.

Die folgenden MSI-Beendigungscodes gelten als erfolgreich:

- 0 (Erfolg)
- 1605 (ERROR_UNKNOWN_PRODUCT)
- 1614 (ERROR_PRODUCT_UNINSTALLED)
- 1641 (Neustart initiiert)
- 3010 (Neustart erforderlich)

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`path`	Geben Sie den Speicherort der MSI-Datei mit einer der folgenden Optionen an:	String	Ja	–	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	• Der lokale Dateipfad. Der Pfad kann absolute oder relative sein. • Ein gültiger S3-Objekt-URI. • Eine gültige Web-HTTP/HTTPS-URL (HTTPS wird empfohlen), die dem RFC-3986-Standard entspricht. Verkettungsausdrücke sind zulässig.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`reboot`	Konfiguriert das Systemneustartverhalten, das auf eine erfolgreiche Ausführung des Aktionsmoduls folgt. Einstellungen: • `Force` – Initiiert einen Systemneustart, nachdem der msiexec Befehl erfolgreich ausgeführt wurde. • `Allow` – Startet einen Systemneustart,	String	Nein	`Allow`	`Allow`, `Force`, `Skip`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	wenn der msiexec Befehl einen Beendigungscode zurückgibt, der angibt, dass ein Neustart erforderlich ist. • `Skip` – Protokolliert eine Informationsmeldung in der `console.log` Datei, die angibt, dass ein Neustart übersprungen wurde. Diese Option verhindert einen Neustart, auch wenn der msiexec				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Befehl einen Beendigungscode zurückgibt, der angibt, dass ein Neustart erforderlich ist.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logOptions`	Geben Sie die Optionen an, die für die MSI-Installationsprotokollierung verwendet werden sollen. Angegebene Flags werden zusammen mit dem `/L` Befehlszeilenparameter an das MSI-Installationsprogramm übergeben, um die Protokollierung zu aktivieren. Wenn keine Flags angegeben sind, AWSTOE verwendet den Standardwert.	String	Nein	`*VX`	`i,w,e,a,r,u,c,m,o,p,v,x,+,!,*`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Weitere Informationen zu Protokoll optionen für MSI finden Sie unter Befehlsze ilenoptio nen in der Produktdo kumentati on zum Microsoft Windows Installer.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logFile`	Ein absoluter oder relativer Pfad zum Speicherort der Protokoll datei. Wenn der Protokoll dateipfad nicht vorhanden ist, wird er erstellt. Wenn der Protokoll dateipfad nicht angegeben wird, speichert das MSI-Installationsp rotokoll AWSTOE nicht.	String	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`properties`	Schlüssel-Wert-Paare der MSI-Protokollierungseigenschaft, zum Beispiel: `TARGETDIR: "C:\target\location"` Hinweis: Die Änderung der folgenden Eigenschaften ist nicht zulässig: • `REBOOT="ReallySupress"` • `REINSTALLMODE="ecmus"` • `REINSTALL="ALL"`	Map[Zeichenfolge]Zeichenfolge	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`ignoreAuthenticodeSignatureErrors`	Flag, um Fehler bei der Validierung der Authentifizierungssignatur für das im Pfad angegebene Installationsprogramm zu ignorieren. Der Get-AuthenticodeSignature Befehl wird verwendet, um die Installationsprogramme zu validieren. Einstellungen: • `true` – Validierungsfehler werden ignoriert und das	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Installationsprogramm wird ausgeführt. • `false` – Validierungsfehler werden nicht ignoriert. Das Installationsprogramm wird nur ausgeführt, wenn die Validierung erfolgreich ist. Dies ist das Standardverhalten.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`allowUnsignedInstaller`	Flag, um die Ausführung des im Pfad angegebenen unsignierten Installationsprogramms zu ermöglichen. Der Get-AuthenticodeSignature Befehl wird verwendet, um die Installationsprogramme zu validieren. Einstellungen: • `true` – Ignoriert den vom Get-AuthenticodeSignature Befehl zurückgegebenen	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	`NotSigned` Status und führt das Installationsprogramm aus. • `false` – Erfordert, dass das Installationsprogramm signiert ist. Unsignierte Installationsprogramme werden nicht ausgeführt. Dies ist das Standardverhalten.				

Beispiele

Die folgenden Beispiele zeigen je nach Installationspfad Variationen des Eingabeabschnitts für Ihr Komponentendokument.

Eingabebeispiel: Installation des lokalen Dokumentpfads entfernen

```
- name: local-path-uninstall
  steps:
```

```
    - name: LocalPathUninstaller
      action: UninstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-uninstall.log
        logOptions: '*VX'
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

Eingabebeispiel: Entfernen der Amazon S3-Pfadinstallation

```
- name: s3-path-uninstall
  steps:
    - name: S3PathUninstaller
      action: UninstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-uninstall.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

Eingabebeispiel: Webpfadinstallation entfernen

```
- name: web-path-uninstall
  steps:
    - name: WebPathUninstaller
      action: UninstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-uninstall.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: false
```

Output

Im Folgenden finden Sie ein Beispiel für die Ausgabe des -`UninstallMSI`Aktionsmoduls.

```
{
```

```
  "logFile": "web-path-uninstall.log",
  "msiExitCode": 0,
  "stdout": ""
}
```

Module für Systemaktionen

Im folgenden Abschnitt werden Aktionsmodule beschrieben, die Befehle und Anweisungen für Dateisystemaktionen ausführen.

Module für Systemaktionen

- Neustart
- SetRegistry
- UpdateOS

Neustart

Das Neustartaktionsmodul startet die Instance neu. Es verfügt über eine konfigurierbare Option, um den Start des Neustarts zu verzögern. Standardmäßig `delaySeconds` ist auf gesetzt`0`, was bedeutet, dass es keine Verzögerung gibt. Das Schritt-Timeout wird für das Aktionsmodul Neustart nicht unterstützt, da es nicht gilt, wenn die Instance neu gestartet wird.

Wenn die Anwendung vom Systems Manager Agent aufgerufen wird, übergibt sie den Beendigungscode (`3010` für Windows, `194` für Linux) an den Systems Manager Agent. Der Systems Manager Agent verarbeitet den Systemneustart wie unter Neustarten verwalteter Instances von Skripts beschrieben.

Wenn die Anwendung auf dem Host als eigenständiger Prozess aufgerufen wird, speichert sie den aktuellen Ausführungsstatus, konfiguriert einen automatischen Ausführungsauslöser nach dem Neustart, um die Anwendung nach dem Neustart neu auszuführen, und startet dann das System neu.

Auslöser für die automatische Ausführung nach dem Neustart:

- Windows . AWSTOE erstellt einen Windows Task Scheduler-Eintrag mit einem Auslöser, der automatisch unter ausgeführt wird `SystemStartup`
- Linux . AWSTOE fügt einen Auftrag in Crontab hinzu, der nach dem Neustart des Systems automatisch ausgeführt wird.

```
@reboot /download/path/awstoe run --document s3://bucket/key/doc.yaml
```

Dieser Auslöser wird beim Start der Anwendung bereinigt.

> **Wiederholversuche**
>
> Standardmäßig ist die maximale Anzahl von Wiederholungen auf den Systems Manager festgelegt`CommandRetryLimit`. Wenn die Anzahl der Neustarts das Wiederholungslimit überschreitet, schlägt die Automatisierung fehl. Sie können das Limit ändern, indem Sie die Systems Manager Agent-Konfigurationsdatei (`Mds.CommandRetryLimit`) bearbeiten. Weitere Informationen finden Sie unter Laufzeitkonfiguration in der Open Source des Systems-Manager-Agenten.

Um das Aktionsmodul Neustart zu verwenden, müssen Sie für Schritte, die einen Neustart enthalten `exitcode` (z. B. `3010`), die Anwendungsbinärdatei als ausführen`sudo user`.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`delaySeconds`	Verzögert eine bestimmte Zeit, bevor ein Neustart eingeleitet wird.	Ganzzahl	Nein	`0`

Eingabebeispiel: Neustartschritt

```
name: RebootStep
action: Reboot
onFailure: Abort
maxAttempts: 2
inputs:
    delaySeconds: 60
```

Ausgabe

Keine.

Wenn das Neustartmodul abgeschlossen ist, fährt Image Builder mit dem nächsten Schritt im Build fort.

SetRegistry

Das SetRegistry Aktionsmodul akzeptiert eine Liste von Eingaben und ermöglicht es Ihnen, den Wert für den angegebenen Registrierungsschlüssel festzulegen. Wenn ein Registrierungsschlüssel nicht vorhanden ist, wird er im definierten Pfad erstellt. Diese Funktion gilt nur für Windows.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`path`	Pfad des Registrierungsschlüssels.	String	Ja
`name`	Name des Registrierungsschlüssels.	String	Ja
`value`	Wert des Registrierungsschlüssels.	Zeichenfolge/Nummer/Array	Ja
`type`	Werttyp des Registrierungsschlüssels.	String	Ja

Unterstützte Pfadpräfixe

- `HKEY_CLASSES_ROOT / HKCR:`
- `HKEY_USERS / HKU:`
- `HKEY_LOCAL_MACHINE / HKLM:`
- `HKEY_CURRENT_CONFIG / HKCC:`
- `HKEY_CURRENT_USER / HKCU:`

Unterstützte -Typen

- `BINARY`
- `DWORD`

- QWORD
- SZ
- EXPAND_SZ
- MULTI_SZ

Eingabebeispiel: Festlegen von Registrierungsschlüsselwerten

```
name: SetRegistryKeyValues
action: SetRegistry
maxAttempts: 3
inputs:
    - path: HKLM:\SOFTWARE\MySoftWare
      name: MyName
      value: FirstVersionSoftware
      type: SZ
    - path: HKEY_CURRENT_USER\Software\Test
      name: Version
      value: 1.1
      type: DWORD
```

Ausgabe

Keine.

UpdateOS

Das UpdateOS-Aktionsmodul bietet Unterstützung für die Installation von Windows- und Linux-Updates. Standardmäßig werden alle verfügbaren Updates installiert. Alternativ können Sie eine Liste mit einem oder mehreren spezifischen Updates für das zu installierende Aktionsmodul konfigurieren. Sie können auch Updates angeben, die von der Installation ausgeschlossen werden sollen.

Wenn sowohl die Listen „Include“ als auch „Exclude“ bereitgestellt werden, kann die resultierende Liste der Updates nur die Aktualisierungen enthalten, die in der Liste „Include“ aufgeführt sind und nicht in der Liste „Exclude“ aufgeführt sind.

> **Note**
>
> UpdateOS unterstützt Amazon Linux 2023 (AL2023) nicht. Wir empfehlen Ihnen, Ihr Basis-AMI auf die neue Version zu aktualisieren, die in jeder Version enthalten ist. Weitere

Alternativen finden Sie unter Steuern der Updates, die von Haupt- und Nebenversionen empfangen wurden im Benutzerhandbuch für Amazon Linux 2023.

- Windows . Updates werden von der auf dem Zielcomputer konfigurierten Aktualisierungsquelle installiert.
- Linux . Die Anwendung sucht auf den unterstützten Paketmanager in der Linux-Plattform und verwendet entweder `yum` oder `apt-get` Paketmanager. Wenn keine unterstützt wird, wird ein Fehler zurückgegeben. Sie sollten `sudo` über Berechtigungen zum Ausführen des UpdateOS-Aktionsmoduls verfügen. Wenn Sie keine `sudo` Berechtigungen haben, `error.Input` wird ein zurückgegeben.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`include`	Für Windows können Sie Folgendes angeben: • Eine oder mehrere Microsoft Knowledge Base (KB)-Artikel-IDs, die in die Liste der Updates aufgenommen werden sollen, die möglicherweise installiert werden. Gültige Formate sind `KB1234567` oder `1234567`. • Ein Aktualisierungsname mit einem Platzhalterwert (`*`). Gültige	Zeichenfolgenliste	Nein

Primitiv	Beschreibung	Typ	Erforderlich
	Formate sind `Security*` oder `*Security*` . Für Linux können Sie ein oder mehrere Pakete angeben, die in die Liste der Updates für die Installation aufgenommen werden sollen.		

Primitiv	Beschreibung	Typ	Erforderlich
`exclude`	Für Windows können Sie Folgendes angeben: • Eine oder mehrere Artikel-IDs der Microsoft Knowledge Base (KB), die in die Liste der Updates aufgenommen werden sollen, die von der Installation ausgeschlossen werden sollen. Gültige Formate sind `KB1234567` oder `1234567`. • Ein Aktualisierungsname mit einem Platzhalterwert (`*`). Gültige Formate sind: `Security*` oder `*Security*` . Für Linux können Sie ein oder mehrere Pakete angeben, die aus der Liste der Updates für die Installation ausgeschlossen werden sollen.	Zeichenfolgenliste	Nein

Eingabebeispiel: Unterstützung für die Installation von Linux-Updates hinzufügen

```
name: UpdateMyLinux
action: UpdateOS
onFailure: Abort
maxAttempts: 3
inputs:
    exclude:
        - ec2-hibinit-agent
```

Eingabebeispiel: Unterstützung für die Installation von Windows-Updates hinzufügen

```
name: UpdateWindowsOperatingSystem
action: UpdateOS
onFailure: Abort
maxAttempts: 3
inputs:
  include:
    - KB1234567
    - '*Security*'
```

Ausgabe

Keine.

Konfigurieren der Eingabe für den AWSTOE Run-Befehl

Um die Befehlszeileneingabe für Ihren AWSTOE run Befehl zu optimieren, können Sie Einstellungen für Befehlsparameter und Optionen in eine Konfigurationsdatei im JSON-Format mit einer `.json` Dateierweiterung aufnehmen. AWSTOE kann Ihre Datei von einem der folgenden Speicherorte lesen:

- Ein lokaler Dateipfad (*`./config.json`*).
- Ein S3-Bucket (*`s3://<bucket-path>/<bucket-name>/config.json`*).

Wenn Sie den run Befehl eingeben, können Sie die Eingabekonfigurationsdatei mit dem --config Parameter angeben. Beispielsweise:

```
awstoe run --config <file-path>/config.json
```

Eingabekonfigurationsdatei

Die JSON-Eingabekonfigurationsdatei enthält Schlüssel-Wert-Paare für alle Einstellungen, die Sie direkt über run Befehlsparameter und Optionen bereitstellen können. Wenn Sie sowohl in der Eingabekonfigurationsdatei als auch im run Befehl als Parameter oder Option eine Einstellung angeben, gelten die folgenden Rangregeln:

Vorrangregeln

1. Eine Einstellung, die über einen Parameter oder AWS CLIeine Option direkt dem run Befehl in der bereitgestellt wird, überschreibt jeden Wert, der in der Eingabekonfigurationsdatei für dieselbe Einstellung definiert ist.
2. Eine Einstellung in der Eingabekonfigurationsdatei überschreibt einen Standardwert für die Komponente.
3. Wenn keine anderen Einstellungen an das Komponentendokument übergeben werden, kann ein Standardwert angewendet werden, falls vorhanden.

Es gibt zwei Ausnahmen von dieser Regel: Dokumente und Parameter. Diese Einstellungen funktionieren in der Eingabekonfiguration und als Befehlsparameter unterschiedlich. Wenn Sie die Eingabekonfigurationsdatei verwenden, dürfen Sie diese Parameter nicht direkt für den run Befehl angeben. Dadurch wird ein Fehler generiert.

Komponenteneinstellungen

Die Eingabekonfigurationsdatei enthält die folgenden Einstellungen. Um Ihre Datei zu optimieren, können Sie alle optionalen Einstellungen weglassen, die nicht benötigt werden. Alle Einstellungen sind optional, sofern nicht anders angegeben.

- cwIgnoreFailures (Boolean) – Protokollierungsfehler in den CloudWatch Protokollen ignorieren.
- cwLogGroup (Zeichenfolge) – Der `LogGroup` Name für die CloudWatch Protokolle.
- cwLogRegion (Zeichenfolge) – Die AWS Region, die für die CloudWatch Protokolle gilt.
- cwLogStream (Zeichenfolge) – Der `LogStream` Name für die CloudWatch Protokolle, der anweist, AWSTOE wohin die `console.log` Datei gestreamt werden soll.
- documentS3BucketOwner (Zeichenfolge) – Die Konto-ID des Bucket-Eigentümers für S3-URI-basierte Dokumente.
- documents (Array von Objekten, erforderlich) – Ein Array von JSON-Objekten, die die YAML-Komponentendokumente darstellen, die der AWSTOE run Befehl ausführt. Es muss mindestens ein Komponentendokument angegeben werden.

Jedes Objekt besteht aus den folgenden Feldern:

- path (Zeichenfolge, erforderlich) – Der Dateispeicherort des YAML-Komponentendokuments. Dies muss einer der folgenden sein:
  - Ein lokaler Dateipfad (`./component-doc-example.yaml`).
  - Ein S3-URI (`s3://bucket/key`).
  - Ein Image Builder-Komponenten-Build-Versions-ARN (arn:aws:imagebuilder:us-west-2:123456789012: component/my-example-component/2021.12.02/1).
- Parameter (Array von Objekten) – Ein Array von Schlüssel-Wert-Paarobjekten, die jeweils einen komponentenspezifischen Parameter darstellen, den der run Befehl beim Ausführen des Komponentendokuments übergibt. Parameter sind für Komponenten optional. Für das Komponentendokument sind möglicherweise Parameter definiert.

  Jedes Objekt besteht aus den folgenden Feldern:

  - Name (Zeichenfolge, erforderlich) – Der Name des Komponentenparameters.
  - value (Zeichenfolge, erforderlich) – Der Wert, der an das Komponentendokument für den benannten Parameter übergeben werden soll.

  Weitere Informationen zu Komponentenparametern finden Sie im Abschnitt Parameter auf der Definieren und Referenzieren von Variablen in AWSTOE Seite .

- executonId (Zeichenfolge) – Dies ist die eindeutige ID, die für die Ausführung des aktuellen run Befehls gilt. Diese ID ist in den Ausgabe- und Protokolldateinamen enthalten, um diese Dateien eindeutig zu identifizieren und sie mit der aktuellen Befehlsausführung zu verknüpfen. Wenn diese Einstellung weggelassen wird, AWSTOE generiert eine GUID.
- logDirectory (Zeichenfolge) – Das Zielverzeichnis, in dem alle Protokolldateien aus dieser Befehlsausführung AWSTOE speichert. Standardmäßig befindet sich dieses Verzeichnis innerhalb des folgenden übergeordneten Verzeichnisses: `TOE_<DATETIME>_<EXECUTIONID>`. Wenn Sie das Protokollverzeichnis nicht angeben, AWSTOE verwendet das aktuelle Arbeitsverzeichnis (`.`).
- logS3BucketName (Zeichenfolge) – Wenn Komponentenprotokolle in Amazon S3 gespeichert sind (empfohlen), AWSTOE lädt die Komponentenanwendungsprotokolle in den S3-Bucket mit dem Namen in diesem Parameter hoch.
- logS3BucketOwner (Zeichenfolge) – Wenn Komponentenprotokolle in Amazon S3 gespeichert sind (empfohlen), ist dies die Besitzerkonto-ID für den Bucket, in dem die Protokolldateien AWSTOEschreibt.

- logS3KeyPrefix (Zeichenfolge) – Wenn Komponentenprotokolle in Amazon S3 gespeichert sind (empfohlen), ist dies das S3-Objektschlüsselpräfix für den Protokollspeicherort im Bucket.
- Parameter (Array von Objekten) – Ein Array von Schlüssel-Wert-Paarobjekten, die Parameter darstellen, die global für alle Komponenten gelten, die in der aktuellen run Befehlsausführung enthalten sind.
  - Name (Zeichenfolge, erforderlich) – Der Name des globalen Parameters.
  - value (Zeichenfolge, erforderlich) – Der Wert, der an alle Komponentendokumente für den benannten Parameter übergeben werden soll.
- Phasen (Zeichenfolge) – Eine durch Komma getrennte Liste, die angibt, welche Phasen aus den YAML-Komponentendokumenten ausgeführt werden sollen. Wenn ein Komponentendokument zusätzliche Phasen enthält, werden diese nicht ausgeführt.
- stateDirectory (Zeichenfolge) – Der Dateipfad, in dem Statusverfolgungsdateien gespeichert werden.
- trace (Boolean) – Aktiviert die ausführliche Protokollierung in der Konsole.

Beispiele

Das folgende Beispiel zeigt eine Eingabekonfigurationsdatei, die die `test` Phasen `build` und für zwei Komponentendokumente ausführt: `sampledoc.yaml`, und `conversation-intro.yaml`. Jedes Komponentendokument hat einen Parameter, der nur für sich selbst gilt, und beide verwenden einen gemeinsamen Parameter. Der `project` Parameter gilt für beide Komponentendokumente.

```
{
   "documents": [
     {
       "path": "<file path>/awstoe/sampledoc.yaml>",
       "parameters": [
         {
           "name": "dayofweek",
           "value": "Monday"
         }
       ]
     },
     {
       "path": "<file path>/awstoe/conversation-intro.yaml>",
       "parameters": [
         {
           "name": "greeting",
```

```
            "value": "Hello, HAL."
          }
        ]
      }
    ],
    "phases": "build,test",
    "parameters": [
      {
        "name": "project",
        "value": "examples"
      }
    ],
    "cwLogGroup": "<log_group_name>",
    "cwLogStream": "<log_stream_name>",
    "documentS3BucketOwner": "<owner_aws_account_number>",
    "executionId": "<id_number>",
    "logDirectory": "<local_directory_path>",
    "logS3BucketName": "<bucket_name_for_log_files>",
    "logS3KeyPrefix": "<key_prefix_for_log_files>",
    "logS3BucketOwner": "<owner_aws_account_number>"
}
```

Von Distributor-Paketen verwaltete Komponenten für Windows

AWS Systems Manager Distributor unterstützt Sie beim Verpacken und Veröffentlichen von Software auf AWS Systems Manager verwalteten Knoten. Sie können Ihre eigene Software verpacken und veröffentlichen oder Distributor verwenden, um AWSvon bereitgestellte Agentensoftwarepakete zu finden und zu veröffentlichen. Weitere Informationen zu Systems Manager Distributor finden Sie unter AWS Systems Manager Distributor im AWS Systems Manager -Benutzerhandbuch.

Verwaltete Komponenten für Distributor

Die folgenden von Image Builder verwalteten Komponenten verwenden AWS Systems Manager Distributor, um Anwendungspakete auf Windows-Instances zu installieren.

- Die `distributor-package-windows` verwaltete Komponente verwendet AWS Systems Manager Distributor, um Anwendungspakete zu installieren, die Sie auf Ihrer Windows-Image-Build-Instance angeben. Informationen zum Konfigurieren von Parametern, wenn Sie diese Komponente in Ihr Rezept aufnehmen, finden Sie unter `distributor-package-windows` Als eigenständige Komponente konfigurieren.

- Die `aws-vss-components-windows` Komponente verwendet AWS Systems Manager Distributor, um das `AwsVssComponents` Paket auf Ihrer Windows-Image-Build-Instance zu installieren. Informationen zum Konfigurieren von Parametern, wenn Sie diese Komponente in Ihr Rezept aufnehmen, finden Sie unter `aws-vss-components-windows` Als eigenständige Komponente konfigurieren.

Weitere Informationen zur Verwendung verwalteter Komponenten in Ihrem Image-Builder-Rezept finden Sie unter Erstellen einer neuen Version eines Image-Rezepts für Image-Rezepte oder Erstellen einer neuen Version eines Container-Rezepts für Container-Rezepte. Weitere Informationen zum -`AwsVssComponents`Paket finden Sie unter Erstellen eines anwendungskonsistenten VSS-Snapshots im Amazon EC2-Benutzerhandbuch für Windows-Instances.

Voraussetzungen

Bevor Sie Image-Builder-Komponenten verwenden, die für die Installation von Anwendungspaketen auf Systems Manager Distributor angewiesen sind, müssen Sie sicherstellen, dass die folgenden Voraussetzungen erfüllt sind.

- Image-Builder-Komponenten, die Systems Manager Distributor zum Installieren von Anwendungspaketen auf Ihrer Instance verwenden, benötigen die Berechtigung zum Aufrufen der Systems-Manager-API. Bevor Sie die Komponenten in einem Image-Builder-Rezept verwenden, müssen Sie die IAM-Richtlinie und -Rolle erstellen, die die Berechtigung erteilen. Informationen zum Konfigurieren von Berechtigungen finden Sie unter Konfigurieren von Berechtigungen für Systems Manager Distributor.

> **Note**
>
> Image Builder unterstützt derzeit keine Systems Manager Distributor-Pakete, die die Instance neu starten. Beispielsweise starten die Pakete `AWSNVMeAWSPVDrivers`, und `AwsEnaNetworkDriver` Distributor die Instance neu und sind daher nicht zulässig.

Konfigurieren von Berechtigungen für Systems Manager Distributor

Die `distributor-package-windows` Komponente und andere Komponenten, die sie verwenden, z. B. `aws-vss-components-windows`, benötigen zusätzliche Berechtigungen für die Ausführung

der Build-Instance. Die Build-Instance muss in der Lage sein, die Systems Manager-API aufzurufen, um eine Distributor-Installation zu starten und das Ergebnis abzufragen.

Befolgen Sie diese Verfahren in der , AWS Management Console um eine benutzerdefinierte IAM-Richtlinie und -Rolle zu erstellen, die Image Builder-Komponenten die Berechtigung zum Installieren von Systems Manager Distributor-Paketen von der Build-Instance gewährt.

Schritt 1: Erstellen einer Richtlinie

Erstellen Sie eine IAM-Richtlinie für Distributor-Berechtigungen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Policies (Richtlinien) und dann Create policy (Richtlinie erstellen).
3. Wählen Sie auf der Seite Richtlinie erstellen die Registerkarte JSON aus und ersetzen Sie dann den Standardinhalt durch die folgende JSON-Richtlinie, indem Sie bei Bedarf Partition, Region und Konto-ID ersetzen oder Platzhalter verwenden.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowDistributorSendCommand",
   "Effect": "Allow",
   "Action": [
    "ssm:SendCommand"
   ],
   "Resource": [
    "arn:${AWS::Partition}:ssm:${AWS::Region}::document/AWS-ConfigureAWSPackage",
    "arn:${AWS::Partition}:ec2:${AWS::Region}:${AWS::AccountId}:instance/*"
   ]
  },
  {
   "Sid": "AllowGetCommandInvocation",
   "Effect": "Allow",
   "Action": [
    "ssm:GetCommandInvocation"
   ],
   "Resource": [
    "*"
   ]
  }
```

```
 ]
}
```

4. Wählen Sie Review policy (Richtlinie prüfen) aus.
5. Geben Sie für Name einen Namen zum Identifizieren der Richtlinie ein, wie z. B. *InvokeDistributor* oder einen anderen von Ihnen bevorzugten Namen.
6. (Optional) Geben Sie für Description (Beschreibung) eine Beschreibung des Zwecks der Rolle ein.
7. Klicken Sie auf Create Policy.

Schritt 2: Erstellen einer Rolle

Erstellen Sie eine IAM-Rolle für Distributor-Berechtigungen.

1. Wählen Sie im Navigationsbereich der IAM-Konsole Rollen und dann Rolle erstellen aus.
2. Wählen Sie unter Select type of trusted entity (Typ der vertrauenswürdigen Entität auswählen) die Option AWS-Service Service aus.
3. Wählen Sie für Choose the service that will use this role (Wählen Sie den Service aus, der diese Rolle verwendet) die Option EC2 und danach Next: Permissions (Nächster Schritt: Berechtigungen) aus.
4. Wählen Sie unter Select your use case (Anwendungsfall auswählen) die Option EC2 und anschließend Next: Permissions (Weiter: Berechtigungen) aus.
5. Aktivieren Sie in der Richtlinienliste das Kontrollkästchen neben AmazonSSMManagedInstanceCore . (Geben Sie `SSM` in das Suchfeld ein, wenn Sie die Liste eingrenzen müssen.)
6. Aktivieren Sie in dieser Richtlinienliste das Kontrollkästchen neben EC2InstanceProfileForImageBuilder. (Geben Sie `ImageBuilder` in das Suchfeld ein, wenn Sie die Liste eingrenzen müssen.)
7. Wählen Sie Next: Tags (Weiter: Tags (Markierungen)) aus.
8. (Optional) Fügen Sie ein oder mehrere Tag-Schlüsselwertpaare hinzu, um den Zugriff für diese Rolle zu organisieren, zu verfolgen oder zu steuern, und wählen Sie dann Weiter: Überprüfen aus.
9. Geben Sie unter Role name (Rollenname) einen Namen für die Rolle, wie z. B. *InvokeDistributor* oder einen anderen von Ihnen bevorzugten Namen ein.

10. (Optional) Ersetzen Sie für Role description (Rollenbeschreibung) den Standardtext mit einer Beschreibung des Zwecks der Rolle.
11. Wählen Sie Create role (Rolle erstellen) aus. Das System leitet Sie zur Seite Roles (Rollen) zurück.

Schritt 3: Anfügen der Richtlinie an die Rolle

Der letzte Schritt zum Einrichten Ihrer Distributor-Berechtigungen besteht darin, die IAM-Richtlinie an die IAM-Rolle anzuhängen.

1. Wählen Sie auf der Seite Rollen in der IAM-Konsole die Rolle aus, die Sie gerade erstellt haben. Die Seite mit der Rollenzusammenfassung wird geöffnet.
2. Wählen Sie Attach Policies (Richtlinien hinzufügen) aus.
3. Suchen Sie nach der Richtlinie, die Sie im vorherigen Verfahren erstellt haben, und aktivieren Sie das Kontrollkästchen neben dem Namen.
4. Wählen Sie Richtlinie anfügen aus.

Verwenden Sie diese Rolle in der Infrastrukturkonfigurationsressource von Image Builder für jedes Image, das Komponenten enthält, die Systems Manager Distributor verwenden. Weitere Informationen finden Sie unter Erstellen einer Infrastrukturkonfiguration.

`distributor-package-windows` Als eigenständige Komponente konfigurieren

Um die `distributor-package-windows` Komponente in einem Rezept zu verwenden, legen Sie die folgenden Parameter fest, die das zu installierende Paket konfigurieren.

> **Note**
>
> Bevor Sie die `distributor-package-windows` Komponente in einem Rezept verwenden, müssen Sie sicherstellen, dass alle erfüllt Voraussetzungen sind.

- Aktion (erforderlich) – Geben Sie an, ob das Paket installiert oder deinstalliert werden soll. Gültige Werte sind `Install` und `Uninstall`. Der Wert ist standardmäßig `Install`.

- PackageName (Erforderlich) – Der Name des zu installierenden oder zu deinstallierenden Distributor-Pakets. Eine Liste der gültigen Paketnamen finden Sie unter Finden von Distributor-Paketen.
- PackageVersion (Optional) – Die Version des zu installierenden Distributor-Pakets. PackageVersion Standards ist die empfohlene Version.
- AdditionalArguments (Optional) – Eine JSON-Zeichenfolge, die die zusätzlichen Parameter enthält, die Ihrem Skript zur Installation, Deinstallation oder Aktualisierung eines Pakets zur Verfügung gestellt werden sollen. Weitere Informationen finden Sie unter additionalArguments im Abschnitt aws:configurePackage Inputs auf der Referenzseite des Systems Manager Command document plugin.

`aws-vss-components-windows` Als eigenständige Komponente konfigurieren

Wenn Sie die `aws-vss-components-windows` Komponente in einem Rezept verwenden, können Sie optional den `PackageVersion` Parameter so festlegen, dass er eine bestimmte Version des `AwsVssComponents` Pakets verwendet. Wenn Sie diesen Parameter weglassen, verwendet die Komponente standardmäßig die empfohlene Version des `AwsVssComponents` Pakets.

> **Note**
>
> Bevor Sie die `aws-vss-components-windows` Komponente in einem Rezept verwenden, müssen Sie sicherstellen, dass alle erfüllt Voraussetzungen sind.

Finden von Distributor-Paketen

Amazon und Drittanbieter stellen öffentliche Pakete bereit, die Sie mit Systems Manager Distributor installieren können.

Um die verfügbaren Pakete in der anzuzeigenAWS Management Console, melden Sie sich bei der -AWS Systems ManagerKonsole an und wählen Sie im Navigationsbereich Distributor aus. Auf der Seite Distributor werden alle Pakete angezeigt, die Ihnen zur Verfügung stehen. Weitere Informationen zum Auflisten verfügbarer Pakete mit der AWS CLIfinden Sie unter Anzeigen von Paketen (Befehlszeile) im AWS Systems Manager -Benutzerhandbuch.

Sie können auch Ihre eigenen privaten Systems Manager Distributor-Pakete erstellen. Weitere Informationen finden Sie unter Erstellen eines Pakets im AWS Systems Manager - Benutzerhandbuch.

CIS-Harding-Komponenten

Das Center for Internet Security (CIS) ist eine Community-gesteuerte Non-Prominente-Organisation. Ihre Cybersicherheitsexperten arbeiten zusammen, um IT-Sicherheitsrichtlinien zu entwickeln, die öffentliche und private Organisationen vor Cyberbedrohungen schützen. Ihre global anerkannte Reihe von bewährten Methoden, CIS Benchmarks, hilft IT-Organisationen auf der ganzen Welt bei der sicheren Konfiguration ihrer Systeme. Trendartikel, Blogbeiträge, Podcasts, Webinare und Whitepaper finden Sie unter CIS Insights auf der Website des Center for Internet Security.

CIS Benchmarks

CIS erstellt und verwaltet eine Reihe von Konfigurationsrichtlinien, die als CIS Benchmarks bezeichnet werden und bewährte Methoden für die Konfiguration bestimmter Technologien bieten, einschließlich Betriebssystemen, Cloud-Plattformen, Anwendungen, Datenbanken und mehr. CIS-Benchmarks werden von Organisationen und Standards wie PCI DSS, HIPAA, DoD Cloud Computing SRG, FISMA, DFARS und FEDRAMP als Branchenstandard anerkannt. Weitere Informationen finden Sie unter CIS-Benchmarks auf der Website des Center for Internet Security.

CIS-Harding-Komponenten

Wenn Sie ein CIS-gehärtetes Image in abonnierenAWS Marketplace, erhalten Sie auch Zugriff auf die zugehörige Hardening-Komponente, die ein Skript ausführt, um CIS Benchmarks Level 1-Richtlinien für Ihre Konfiguration durchzusetzen. Die CIS-Organisation besitzt und verwaltet CIS-Hardening-Komponenten, um sicherzustellen, dass sie den neuesten Richtlinien entsprechen.

> **Note**
>
> CIS-Hardening-Komponenten folgen nicht den Standardregeln für die Komponentenreihenfolge in Image-Builder-Rezepten. Die CIS-Harding-Komponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests mit Ihrem Ausgabe-Image ausgeführt werden.

Von Amazon verwaltete STIG-Harding-Komponenten für EC2 Image Builder

Security Technical Implementation Guides (STIGs) sind die Standards zur Konfigurationsverdichtung, die von der US-Verteidigungsbehörde für Informationssysteme (DISA) zur Sicherung von Informationssystemen und Software erstellt wurden. Um Ihre Systeme mit STIG-Standards konform zu machen, müssen Sie eine Vielzahl von Sicherheitseinstellungen installieren, konfigurieren und testen.

Image Builder bietet STIG-Harding-Komponenten, mit denen Sie effizienter konforme Images für grundlegende STIG-Standards erstellen können. Diese STIG-Komponenten scannen auf Fehlkonfigurationen und führen ein Korrekturskript aus. Für die Verwendung von STIG-konformen Komponenten fallen keine zusätzlichen Gebühren an.

> **Important**
>
> Mit wenigen Ausnahmen installieren STIG-Harding-Komponenten keine Pakete von Drittanbietern. Wenn Pakete von Drittanbietern bereits auf der Instance installiert sind und verwandte STIGs vorhanden sind, die Image Builder für dieses Paket unterstützt, wendet die härtende Komponente sie an.

Auf dieser Seite werden alle STIGs aufgeführt, die Image Builder unterstützt und die auf die EC2-Instances angewendet werden, die Image Builder startet, wenn Sie ein neues Image erstellen und testen. Wenn Sie zusätzliche STIG-Einstellungen auf Ihr Image anwenden möchten, können Sie eine benutzerdefinierte Komponente erstellen, um sie zu konfigurieren. Weitere Informationen zu benutzerdefinierten Komponenten und deren Erstellung finden Sie unter Verwalten von Komponenten mit Image Builder.

Wenn Sie ein Image erstellen, protokollieren die STIG-Hardening-Komponenten, ob unterstützte STIGs angewendet oder übersprungen werden. Wir empfehlen Ihnen, die Image-Builder-Protokolle für Ihre Images zu überprüfen, die STIG-Härterkomponenten verwenden. Weitere Informationen zum Zugriff auf und Überprüfen von Image-Builder-Protokollen finden Sie unter Fehlerbehebung bei Pipeline-Builds.

Compliance-Stufen

- Hoch (Kategorie I)

Das schwerwiegendste Risiko. Schließt jede Schwachstelle ein, die zu einem Verlust der Vertraulichkeit, Verfügbarkeit oder Integrität führen kann.

- Mittel (Kategorie II)

  Beinhaltet jede Schwachstelle, die zu einem Verlust der Vertraulichkeit, Verfügbarkeit oder Integrität führen kann, aber die Risiken können verringert werden.

- Niedrig (Kategorie III)

  Jede Schwachstelle, die Maßnahmen zum Schutz vor einem Verlust der Vertraulichkeit, Verfügbarkeit oder Integrität beeinträchtigt.

Themen

Windows-STIG-Harding-Komponenten

AWSTOE Windows-STIG-Hardening-Komponenten sind für eigenständige Server konzipiert und wenden die lokale Gruppenrichtlinie an. STIG-konforme Hardening-Komponenten installieren InstallRoot vom US-Verteidigungsministerium (Department of Defense, DoD) auf der Windows-Infrastruktur, um die DoD-Zertifikate herunterzuladen, zu installieren und zu aktualisieren. Sie entfernen auch unnötige Zertifikate, um die STIG-Compliance aufrechtzuerhalten. Derzeit werden STIG-Baselines für die folgenden Versionen von Windows Server unterstützt: 2012 R2, 2016, 2019 und 2022.

In diesem Abschnitt werden die aktuellen Einstellungen für jede der Windows-STIG-Hardening-Komponenten aufgeführt, gefolgt von einem Versionsverlaufsprotokoll.

STIG-Build-Windows-Low-Version 2022.4.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht anwendbar ist, überspringt

die Hardening-Komponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Organisationsspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Hardening-Komponente anwendet, z. B. wenn Administratoren Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste der Windows-STIGs finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

- Windows Server 2022 STIG Version 1 Version 1

  V-254335, V-254336, V-254337, V-254338, V-254351, V-254357, V-254363 und V-254481
- Windows Server 2019 STIG Version 2 Version 5

  V-205691, V-205819, V-205858, V-205859, V-205860, V-205870, V-205871 und V-205923
- Windows Server 2016 STIG Version 2 Version 5

  V-224916, V-224917, V-224918, V-224919, V-224931, V-224942 und V-225060
- Windows Server 2012 R2 MS STIG Version 3 Version 5

  V-225537, V-225536, V-225526, V-225525, V-225514, V-225511, V-225490, V-225489, V-225488, V-225487, V-225485, V-225484, V-225483, V-225482, V-225481, V-225480, V-225479, V-225476, V-225473, V-225468, V-225462, V-225460, V-225459, V-225412, V-225394, V-225392, V-225376, V-225363, V-225362, V-225360, V-225359, V-225358, V-225357, V-225355, V-225343, V-225342, V-225336, V-225335, V-225334, V-225333, V-225332, V-225331, V-225330, V-225328, V-225327, V-225324, V-225319, V-225318 und V-225250
- Microsoft .NET Framework 4.0 STIG Version 2 Version 2

  Auf das Microsoft .NET Framework für Sicherheitslücken der Kategorie III werden keine STIG-Einstellungen angewendet.
- Windows Firewall STIG Version 2 Version 1

  V-241994, V-241995, V-241996, V-241999, V-242000, V-242001, V-242006, V-242007 und V-242008
- Internet Explorer 11 STIG Version 2 Version 3

  V-46477, V-46629 und V-97527
- Microsoft Edge STIG Version 1 Release 6 (nur Windows Server 2022)

  V-235727, V-235731, V-235751, V-235752 und V-235765

STIG-Build-Windows-Medium Version 2022.4.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht anwendbar ist, überspringt die Hardening-Komponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Organisationsspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Hardening-Komponente anwendet, z. B. wenn Administratoren Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste der Windows-STIGs finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die STIG-Build-Windows-Medium-Hardening-Komponenten enthalten alle aufgelisteten STIG-Einstellungen, die für STIG-Build-Windows-Low-Hardening-Komponenten AWSTOE gelten, zusätzlich zu den STIG-Einstellungen, die speziell für Schwachstellen der Kategorie II aufgeführt sind.

- Windows Server 2022 STIG Version 1 Version 1

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-254247, V-254265, V-254269, V-254270, V-254271, V-254272, V-254273, V-254274, V-254276, V-254277, V-254278, V-254285, V-254286, V-254287, V-254288, V-254289, V-254290, V-254291, V-254292, V-254300, V-254301, V-254302, V-254303, V-254304, V-254305, V-254306, V-254307, V-254308, V-254309, V-254310, V-254311, V-254312, V-254313, V-254314, V-254315, V-254316, V-254317, V-254318, V-254319, V-254320, V-254321, V-254322, V-254323, V-254324, V-254325, V-254326, V-254327, V-254328, V-254329, V-254330, V-254331, V-254332, V-254333, V-254334, V-254339, V-254341, V-254342, V-254344, V-254345, V-254346, V-254347, V-254348, V-254349, V-254350, V-254355, V-254356, V-254358, V-254359, V-254360, V-254361, V-254362, V-254364, V-254365, V-254366, V-254367, V-254368, V-254369, V-254370, V-254371, V-254372, V-254373, V-254375, V-254376, V-254377, V-254379, V-254380, V-254382, V-254383, V-254431, V-254432, V-254433, V-254434, V-254435, V-254436, V-254438, V-254439, V-254442, V-254443, V-254444, V-254445, V-254449, V-254450, V-254451, V-254452, V-254453, V-254454, V-254455, V-254456, V-254459, V-254460, V-254461, V-254462, V-254463, V-254464, V-254468, V-254470, V-254471,

V-254472, V-254473, V-254476, V-254477, V-254478, V-254479, V-254480, V-254482, V-254483, V-254484, V-254485, V-254486, V-254487, V-254488, V-254489, V-254490, V-254493, V-254494, V-254495, V-254497, V-254499, V-254501, V-254502, V-254503, V-254504, V-254505, V-254507, V-254508, V-254509, V-254510, V-254511, und V-254512

- Windows Server 2019 STIG Version 2 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-205625, V-205626, V-205627, V-205629, V-205630, V-205633, V-205634, V-205635, V-205636, V-205637, V-205638, V-205639, V-205643, V-205644, V-205648, V-205649, V-205650, V-205651, V-205652, V-205655, V-205656, V-205659, V-205660, V-205662, V-205671, V-205672, V-205673, V-205675, V-205676, V-205678, V-205679, V-205680, V-205681, V-205682, V-205683, V-205684, V-205685, V-205686, V-205687, V-205688, V-205689, V-205690, V-205692, V-205693, V-205694, V-205697, V-205698, V-205708, V-205709, V-205712, V-205714, V-205716, V-205717, V-205718, V-205719, V-205720, V-205722, V-205729, V-205730, V-205733, V-205747, V-205751, V-205752, V-205754, V-205756, V-205758, V-205759, V-205760, V-205761, V-205762, V-205764, V-205765, V-205766, V-205767, V-205768, V-205769, V-205770, V-205771, V-205772, V-205773, V-205774, V-205775, V-205776, V-205777, V-205778, V-205779, V-205780, V-205781, V-205782, V-205783, V-205784, V-205795, V-205796, V-205797, V-205798, V-205801, V-205808, V-205809, V-205810, V-205811, V-205812, V-205813, V-205814, V-205815, V-205816, V-205817, V-205821, V-205822, V-205823, V-205824, V-205825, V-205826, V-205827, V-205828, V-205830, V-205832, V-205833, V-205834, V-205835, V-205836, V-205837, V-205838, V-205839, V-205840, V-205841, V-205861, V-205863, V-205865, V-205866, V-205867, V-205868, V-205869, V-205872, V-205873, V-205874, V-205911, V-205912, V-205915, V-205916, V-205917, V-205918, V-205920, V-205921, V-205922, V-205924, V-205925, und V-236001

- Windows Server 2016 STIG Version 2 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-224850, V-224852, V-224853, V-224854, V-224855, V-224856, V-224857, V-224858, V-224859, V-224866, V-224867, V-224868, V-224869, V-224870, V-224871, V-224872, V-224873, V-224881, V-224882, V-224883, V-224884, V-224885, V-224886, V-224887, V-224888, V-224889, V-224890, V-224891, V-224892, V-224893, V-224894, V-224895, V-224896, V-224897, V-224898, V-224899, V-224900, V-224901, V-224902, V-224903, V-224904, V-224905, V-224906, V-224907, V-224908, V-224909, V-224910, V-224911, V-224912, V-224913, V-224914, V-224915, V-224920, V-224922,

V-224924, V-224925, V-224926, V-224927, V-224928, V-224929, V-224930, V-224935, V-224936, V-224937, V-224938, V-224939, V-224940, V-224941, V-224943, V-224944, V-224945, V-224946, V-224947, V-224948, V-224949, V-224951, V-224952, V-224953, V-224955, V-224956, V-224957, V-224959, V-224960, V-224962, V-224963, V-225010, V-225013, V-225014, V-225015, V-225016, V-225017, V-225018, V-225019, V-225021, V-225022, V-225023, V-225024, V-225028, V-225029, V-225030, V-225031, V-225032, V-225033, V-225034, V-225035, V-225038, V-225039, V-225040, V-225041, V-225042, V-225043, V-225047, V-225049, V-225050, V-225051, V-225052, V-225055, V-225056, V-225057, V-225058, V-225061, V-225062, V-225063, V-225064, V-225065, V-225066, V-225067, V-225068, V-225069, V-225072, V-225073, V-225074, V-225076, V-225078, V-225080, V-225081, V-225082, V-225083, V-225084, V-225086, V-225087, V-225088, V-225089, V-225092, V-225093 und V-236000

- Windows Server 2012 R2 MS STIG Version 3 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-225574, V-225573, V-225572, V-225571, V-225570, V-225569, V-225568, V-225567, V-225566, V-225565, V-225564, V-225563, V-225562, V-225561, V-225560, V-225559, V-225558, V-225557, V-225555, V-225554, V-225553, V-225551, V-225550, V-225549, V-225548, V-225546, V-225545, V-225544, V-225543, V-225542, V-225541, V-225540, V-225539, V-225538, V-225535, V-225534, V-225533, V-225532, V-225531, V-225530, V-225529, V-225528, V-225527, V-225524, V-225523, V-225522, V-225521, V-225520, V-225519, V-225518, V-225517, V-225516, V-225515, V-225513, V-225510, V-225509, V-225508, V-225506, V-225504, V-225503, V-225502, V-225501, V-225500, V-225494, V-225486, V-225478, V-225477, V-225475, V-225474, V-225472, V-225471, V-225470, V-225469, V-225464, V-225463, V-225461, V-225458, V-225457, V-225456, V-225455, V-225454, V-225453, V-225452, V-225448, V-225443, V-225442, V-225441, V-225415, V-225414, V-225413, V-225411, V-225410, V-225409, V-225408, V-225407, V-225406, V-225405, V-225404, V-225402, V-225401, V-225400, V-225398, V-225397, V-225395, V-225393, V-225391, V-225389, V-225386, V-225385, V-225384, V-225383, V-225382, V-225381, V-225380, V-225379, V-225378, V-225377, V-225375, V-225374, V-225373, V-225372, V-225371, V-225370, V-225369, V-225368, V-225367, V-225356, V-225353, V-225352, V-225351, V-225350, V-225349, V-225348, V-225347, V-225346, V-225345, V-225344, V-225341, V-225340, V-225339, V-225338, V-225337, V-225329, V-225326, V-225325, V-225317, V-225316, V-225315, V-225314, V-225305, V-225304, V-225303, V-225302, V-225301, V-225300, V-225299, V-225298, V-225297, V-225296, V-225295, V-225294, V-225293, V-225292, V-225291, V-225290, V-225289, V-225288, V-225287, V-225286, V-225285, V-225284, V-225283, V-225282, V-225281, V-225280, V-225279, V-225278, V-225277, V-225276, V-225275,

V-225273, V-225272, V-225271, V-225270, V-225269, V-225268, V-225267, V-225266, V-225265, V-225264, V-225263, V-225261, V-225260, V-225259, und V-225239

- Microsoft .NET Framework 4.0 STIG Version 2 Version 2

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) sowie V-225238 anwendet

- Windows Firewall STIG Version 2 Version 1

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-241989, V-241990, V-241991, V-241993, V-241998 und V-242003

- Internet Explorer 11 STIG Version 2 Version 3

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-46473, V-46475, V-46481, V-46483, V-46501, V-46507, V-46509, V-46511, V-46513, V-46515, V-46517, V-46521, V-46523, V-46525, V-46543, V-46545, V-46547, V-46549, V-46553, V-46555, V-46573, V-46575, V-46577, V-46579, V-46581, V-46583, V-46587, V-46589, V-46591, V-46593, V-46597, V-46599, V-46601, V-46603, V-46605, V-46607, V-46609, V-46615, V-46617, V-46619, V-46621, V-46625, V-46633, V-46635, V-46637, V-46639, V-46641, V-46643, V-46645, V-46647, V-46649, V-46653, V-46663, V-46665, V-46669, V-46681, V-46685, V-46689, V-46691, V-46693, V-46695, V-46701, V-46705, V-46709, V-46711, V-46713, V-46715, V-46717, V-46719, V-46721, V-46723, V-46725, V-46727, V-46729, V-46731, V-46733, V-46779, V-46781, V-46787, V-46789, V-46791, V-46797, V-46799, V-46801, V-46807, V-46811, V-46815, V-46819, V-46829, V-46841, V-46847, V-46849, V-46853, V-46857, V-46859, V-46861, V-46865, V-46869, V-46879, V-46883, V-46885, V-46889, V-46893, V-46895, V-46897, V-46903, V-46907, V-46921, V-46927, V-46939, V-46975, V-46981, V-46987, V-46995, V-46997, V-46999, V-47003, V-47005, V-47009, V-64711, V-64713, V-64715, V-64717, V-64719, V-64721, V-64723, V-64725, V-64729, V-72757, V-72759, V-72761, V-72763, V-75169 und V-75171

- Microsoft Edge STIG Version 1 Release 6 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-235720, V-235721, V-235723, V-235724, V-235725, V-235726, V-235728, V-235729, V-235730, V-235732, V-235733, V-235734, V-235735, V-235736, V-235737, V-235738, V-235739, V-235740,

V-235741, V-235742, V-235743, V-235744, V-235745, V-235746, V-235747, V-235748, V-235749, V-235750, V-235754, V-235756, V-235760, V-235761, V-235763, V-235764, V-235766, V-235767, V-235768, V-235769, V-235770, V-235771, V-235772, V-235773, V-235774 V-2467367

- Defender STIG Version 2 Release 4 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) anwendet, sowie:

  V-213427, V-213429, V-213430, V-213431, V-213432, V-213433, V-213434, V-213435, V-213436, V-213437, V-213438, V-213439, V-213440, V-213441, V-213442, V-213443, V-213444, V-213445, V-213446, V-213447, V-213448, V-213449, V-213450, V-213451, V-213455, V-213464, V-213465 V-2134664

STIG-Build-Windows-High-Version 2022.4.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht anwendbar ist, überspringt die Hardening-Komponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Organisationsspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Hardening-Komponente anwendet, z. B. wenn Administratoren Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste der Windows-STIGs finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die STIG-Build-Windows-High-Hardening-Komponenten umfassen alle aufgelisteten STIG-Einstellungen, die für STIG-Build-Windows-Low- und STIG-Build-Windows-Medium-Hardening-Komponenten AWSTOE gelten, zusätzlich zu den STIG-Einstellungen, die speziell für Schwachstellen der Kategorie I aufgeführt sind.

- Windows Server 2022 STIG Version 1 Version 1

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) anwendet, sowie:

V-254293, V-254352, V-254353, V-254354, V-254374, V-254378, V-254381, V-254446, V-254465, V-254466, V-254467, V-254469, V-254474, V-254475 und V-254500

- Windows Server 2019 STIG Version 2 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) anwendet, sowie:

  V-205653, V-205654, V-205711, V-205713, V-205724, V-205725, V-205757, V-205802, V-205804, V-205805, V-205806, V-205849, V-205908, V-205913, V-205914 und V-205919

- Windows Server 2016 STIG Version 2 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) anwendet, sowie:

  V-224874, V-224932, V-224933, V-224934, V-224954, V-224958, V-224961, V-225025, V-225044, V-225045, V-225046, V-225048, V-225053, V-225054 und V-225079

- Windows Server 2012 R2 MS STIG Version 3 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) anwendet, sowie:

  V-225556, V-225552, V-225547, V-225507, V-225505, V-225498, V-225497, V-225496, V-225493, V-225492, V-225491, V-225449, V-225444, V-225399, V-225396, V-225390, V-225366, V-225365, V-225364, V-225354 und V-225274

- Microsoft .NET Framework 4.0 STIG Version 2 Version 2

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) für das Microsoft .NET Framework anwendet. Für Schwachstellen der Kategorie I gelten keine zusätzlichen STIG-Einstellungen.

- Windows Firewall STIG Version 2 Version 1

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) anwendet, sowie:

  V-241992, V-241997 und V-242002

- Internet Explorer 11 STIG Version 2 Version 3

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für die Schwachstellen der Kategorien II und III (Medium und Low) für Internet Explorer 11 anwendet. Für Schwachstellen der Kategorie I gelten keine zusätzlichen STIG-Einstellungen.

- Microsoft Edge STIG Version 1 Release 6 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) anwendet, sowie:

  V-235758 und V-235759
- Defender STIG Version 2 Release 4 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) anwendet, sowie:

  V-213426, V-213452 und V-213453

STIG-Versionsverlaufsprotokoll für Windows

In diesem Abschnitt wird der Versionsverlauf der Windows-Hardening-Komponenten für die vierteljährlichen STIG-Updates protokolliert. Um die Änderungen und veröffentlichten Versionen für ein Quartal anzuzeigen, wählen Sie den Titel aus, um die Informationen zu erweitern.

Änderungen im Q3 2023 – 10/04/2023 (keine Änderungen):

Für die Veröffentlichung des dritten Quartals 2023 wurden keine Änderungen an der Windows-Komponente STIGS vorgenommen.

Änderungen für Q2 2023 – 05/03/2023 (keine Änderungen):

Für die Veröffentlichung des zweiten Quartals 2023 wurden keine Änderungen für die Windows-Komponente STIGS vorgenommen.

Änderungen für Q1 2023 – 03/27/2023 (keine Änderungen):

Für die Veröffentlichung des ersten Quartals 2023 wurden keine Änderungen an der Windows-Komponente STIGS vorgenommen.

Änderungen für Q4 2022 – 02/01/2023:

Aktualisierte STIG-Versionen und angewandte STIGS für die Q4-Version 2022 wie folgt:

STIG-Build-Windows-Low-Version 2022.4.x

- Windows Server 2022 STIG Version 1 Release 1
- Windows Server 2019 STIG Version 2 Release 5
- Windows Server 2016 STIG Version 2 Release 5
- Windows Server 2012 R2 MS STIG Version 3 Release 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 2
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 2 Release 3
- Microsoft Edge STIG Version 1 Release 6 (nur Windows Server 2022)

STIG-Build-Windows-Medium Version 2022.4.x

- Windows Server 2022 STIG Version 1 Release 1
- Windows Server 2019 STIG Version 2 Release 5
- Windows Server 2016 STIG Version 2 Release 5
- Windows Server 2012 R2 MS STIG Version 3 Release 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 2
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 2 Release 3
- Microsoft Edge STIG Version 1 Release 6 (nur Windows Server 2022)
- Defender STIG Version 2 Release 4 (nur Windows Server 2022)

STIG-Build-Windows-High-Version 2022.4.x

- Windows Server 2022 STIG Version 1 Release 1
- Windows Server 2019 STIG Version 2 Release 5
- Windows Server 2016 STIG Version 2 Release 5
- Windows Server 2012 R2 MS STIG Version 3 Release 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 2
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 2 Release 3

- Microsoft Edge STIG Version 1 Release 6 (nur Windows Server 2022)
- Defender STIG Version 2 Release 4 (nur Windows Server 2022)

Änderungen für Q3 2022 – 09/30/2022 (keine Änderungen):

Für die Veröffentlichung des dritten Quartals 2022 wurden keine Änderungen an der Windows-Komponente STIGS vorgenommen.

Änderungen für Q2 2022 – 08/02/2022:

Aktualisierte STIG-Versionen und angewandte STIGS für die Q2-Version 2022.

STIG-Build-Windows-Low Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 4
- Windows Server 2016 STIG Version 2 Version 4
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-Medium Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 4
- Windows Server 2016 STIG Version 2 Version 4
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-High-Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 4
- Windows Server 2016 STIG Version 2 Version 4
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1

- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

Änderungen für Q1 2022 – 08/02/2022 (keine Änderungen):

Für die Veröffentlichung des ersten Quartals 2022 wurden keine Änderungen an der Windows-Komponente STIGS vorgenommen.

Änderungen für Q4 2021 – 12/20/2021:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des vierten Quartals 2021.

STIG-Build-Windows-Low Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 3
- Windows Server 2016 STIG Version 2 Version 3
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-Medium Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 3
- Windows Server 2016 STIG Version 2 Version 3
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-High-Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 3
- Windows Server 2016 STIG Version 2 Version 3
- Windows Server 2012 R2 MS STIG Version 3 Version 3

- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

Änderungen im Q3 2021 – 09/30/2021:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des dritten Quartals 2021.

STIG-Build-Windows-Low Version 1.4.x

- Windows Server 2019 STIG Version 2 Version 2
- Windows Server 2016 STIG Version 2 Version 2
- Windows Server 2012 R2 MS STIG Version 3 Version 2
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 1 Version 7
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-Medium Version 1.4.x

- Windows Server 2019 STIG Version 2 Version 2
- Windows Server 2016 STIG Version 2 Version 2
- Windows Server 2012 R2 MS STIG Version 3 Version 2
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 1 Version 7
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-High-Version 1.4.x

- Windows Server 2019 STIG Version 2 Version 2
- Windows Server 2016 STIG Version 2 Version 2
- Windows Server 2012 R2 MS STIG Version 3 Version 2
- Microsoft .NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 1 Version 7

- Internet Explorer 11 STIG Version 1 Version 19

Linux STIG-Harding-Komponenten

Dieser Abschnitt enthält Informationen zu Linux-STIG-Hardening-Komponenten, gefolgt von einem Versionsverlaufsprotokoll. Wenn die Linux-Distribution keine eigenen STIG-Einstellungen hat, wendet die Hardening-Komponente RHEL-Einstellungen an. Die Hardening-Komponente wendet unterstützte STIG-Einstellungen auf die Infrastruktur basierend auf der Linux-Distribution wie folgt an:

Red Hat Enterprise Linux (RHEL) 7 STIG-Einstellungen

- RHEL 7
- CentOS 7
- Amazon Linux 2 (AL2)

RHEL-8-STIG-Einstellungen

- RHEL 8
- CentOS 8
- Amazon Linux 2023 (AL 2023)

STIG-Build-Linux-Low-Version 2023.3.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht anwendbar ist, überspringt die Hardening-Komponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Organisationsspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Hardening-Komponente anwendet, z. B. wenn Administratoren Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

RHEL 7 STIG Version 3, Version 12

- RHEL 7/CentOS 7

  V-204452, V-204576 und V-204605

- AL2

  V-204452, V-204576 und V-204605

RHEL 8 STIG Version 1 Version 11

- RHEL 8/CentOS 8/AL 2023

  V-230241, V-230253, V-230269, V-230270, V-230281, V-230285, V-230346, V-230381, V-230395, V-230468, V-230469, V-230485, V-230486, V-230491, V-230494, V-230495, V-230496, V-230497, V-230498, V-2304999949 und V-2445272

Ubuntu 18.04 STIG Version 2 Version 11

V-219163, V-219164, V-219165, V-219172, V-219173, V-219174, V-219175, V-219178, V-219180, V-219210, V-219301, V-219327, V-219332 und V-2193333

Ubuntu 20.04 STIG Version 1 Version 9

V-238202, V-238221, V-238222, V-238223, V-238224, V-238226, V-238234, V-238235, V-238237, V-238308, V-238323, V-238357, V-238362 und V-238373

STIG-Build-Linux-Medium Version 2023.3.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht anwendbar ist, überspringt die Hardening-Komponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Organisationsspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Hardening-Komponente anwendet, z. B. wenn Administratoren Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die STIG-Build-Linux-Medium-Hardening-Komponenten enthalten alle aufgelisteten STIG-Einstellungen, die für STIG-Build-Linux-Low-Hardening-Komponenten AWSTOE gelten,

zusätzlich zu den STIG-Einstellungen, die speziell für Schwachstellen der Kategorie II aufgeführt sind.

RHEL 7 STIG Version 3 Version 12

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) für diese Linux-Distribution anwendet, sowie:

- RHEL 7/CentOS 7

  V-204405, V-204406, V-204407, V-204408, V-204409, V-204410, V-204411, V-204412, V-204413, V-204414, V-204415, V-204416, V-204417, V-204418, V-204422, V-204423, V-204426, V-204427, V-204428, V-204431, V-204435, V-204437, V-204449, V-204450, V-204451, V-204457, V-204466, V-204503, V-204516, V-204517, V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V-204541, V-204542, V-204543, V-204544, V-204545, V-204546, V-204547, V-204548, V-204549, V-204550, V-204551, V-204552, V-204553, V-204554, V-204555, V-204556, V-204557, V-204558, V-204559, V-204560, V-204562, V-204563, V-204564, V-204565, V-204566, V-204567, V-204568, V-204572, V-204579, V-204584, V-204585, V-204586, V-204587, V-204589, V-204590, V-204591, V-204592, V-204593, V-204598, V-204599, V-204600, V-204601, V-204602, V-204609, V-204610, V-204611, V-204612, V-204613, V-204614, V-204615, V-204616, V-204617, V-204619, V-204622, V-204624, V-204625, V-204630, V-204631, V-204633, V-233307, V-237634, V-237635, V-251703, V-255925, V-255927, V-255928, und V-256970

- AL2:

  V-204405, V-204406, V-204407, V-204408, V-204409, V-204410, V-204411, V-204412, V-204413, V-204414, V-204415, V-204416, V-204417, V-204418, V-204422, V-204423, V-204426, V-204427, V-204428, V-204431, V-204435, V-204437, V-204449, V-204450, V-204451, V-204457, V-204466, V-204503, V-204516, V-204517, V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V-204541, V-204542, V-204543, V-204544, V-204545, V-204546, V-204547, V-204548, V-204549, V-204550, V-204551, V-204552, V-204553, V-204554, V-204555, V-204556, V-204557, V-204558, V-204559, V-204560, V-204562, V-204563, V-204564, V-204565, V-204566, V-204567, V-204568, V-204572, V-204578, V-204579, V-204584, V-204585, V-204586, V-204587, V-204589, V-204590, V-204591, V-204592, V-204593, V-204595, V-204598, V-204599, V-204600, V-204601, V-204602, V-204609, V-204610, V-204611, V-204612, V-204613, V-204614, V-204615, V-204616, V-204617, V-204619, V-204622, V-204624, V-204625, V-204630, V-204631, V-204633, V-233307, V-237634, V-237635, V-251703, V-255925, V-255927, V-255928, und V-256970

RHEL 8 STIG Version 1 Version 11

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) für diese Linux-Distribution anwendet, sowie:

- RHEL 8/CentOS 8/AL 2023

  V-230228, V-230231, V-230233, V-230236, V-230237, V-230239, V-230240, V-230244, V-230255, V-230266, V-230267, V-230268, V-230273, V-230275, V-230277, V-230278, V-230280, V-230282, V-230288, V-230290, V-230291, V-230296, V-230298, V-230310, V-230311, V-230312, V-230313, V-230314, V-230315, V-230324, V-230330, V-230332, V-230333, V-230334, V-230335, V-230336, V-230337, V-230338, V-230339, V-230340, V-230341, V-230342, V-230343, V-230344, V-230345, V-230348, V-230349, V-230356, V-230357, V-230358, V-230359, V-230360, V-230361, V-230362, V-230363, V-230365, V-230368, V-230369, V-230370, V-230375, V-230377, V-230378, V-230382, V-230383, V-230386, V-230387, V-230390, V-230392, V-230402, V-230403, V-230404, V-230405, V-230406, V-230407, V-230408, V-230409, V-230410, V-230411, V-230412, V-230413, V-230418, V-230419, V-230421, V-230422, V-230423, V-230424, V-230425, V-230426, V-230427, V-230428, V-230429, V-230430, V-230431, V-230432, V-230433, V-230434, V-230435, V-230436, V-230437, V-230438, V-230439, V-230444, V-230446, V-230447, V-230448, V-230449, V-230455, V-230456, V-230462, V-230463, V-230464, V-230465, V-230466, V-230467, V-230478, V-230480, V-230488, V-230489, V-230502, V-230503, V-230526, V-230527, V-230532, V-230535, V-230536, V-230537, V-230538, V-230539, V-230540, V-230541, V-230542, V-230543, V-230544, V-230545, V-230546, V-230547, V-230548, V-230549, V-230555, V-230556, V-230559, V-230560, V-230561, V-237640, V-237642, V-237643, V-244523, V-244524, V-244525, V-244526, V-244528, V-244533, V-244534, V-244537, V-244542, V-244549, V-244550, V-244551, V-244552, V-244553, V-244554, V-250317, V-251711, V-251713, V-251714, V-251715, V-251716, V-251717, V-251718, und V-256974

Ubuntu 18.04 STIG Version 2 Version 11

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) für diese Linux-Distribution anwendet, sowie:

V-219149, V-219155, V-219156, V-219160, V-219166, V-219176, V-219181, V-219184, V-219186, V-219187, V-219188, V-219189, V-219190, V-219191, V-219192, V-219193, V-219194, V-219195, V-219196, V-219197, V-219198, V-219199, V-219200, V-219201, V-219202, V-219203, V-219204, V-219205, V-219206, V-219207, V-219208, V-219209, V-219213, V-219214, V-219215, V-219216, V-219217, V-219218, V-219219, V-219220, V-219221, V-219222, V-219223, V-219224, V-219227, V-219228, V-219229, V-219230, V-219231, V-219232, V-219233, V-219234, V-219235, V-219236,

V-219238, V-219239, V-219240, V-219241, V-219242, V-219243, V-219244, V-219250, V-219254, V-219257, V-219263, V-219264, V-219265, V-219266, V-219267, V-219268, V-219269, V-219270, V-219271, V-219272, V-219273, V-219274, V-219275, V-219276, V-219277, V-219279, V-219281, V-219287, V-219291, V-219297, V-219298, V-219299, V-219300, V-219303, V-219306, V-219309, V-219310, V-219311, V-219312, V-219315, V-219326, V-219328, V-219330, V-219331, V-219334, V-219335, V-219336, V-219337, V-219338, V-219339, V-219342, V-233779, V-233780, V-237768, V-237769, V-237770, und V-255906

Ubuntu 20.04 STIG Version 1 Version 9

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorie III (niedrig) für diese Linux-Distribution anwendet, sowie:

V-238200, V-238205, V-238207, V-238209, V-238211, V-238212, V-238213, V-238216, V-238220, V-238225, V-238227, V-238228, V-238230, V-238231, V-238236, V-238238, V-238239, V-238240, V-238241, V-238242, V-238244, V-238245, V-238246, V-238247, V-238248, V-238249, V-238250, V-238251, V-238252, V-238253, V-238254, V-238255, V-238256, V-238257, V-238258, V-238264, V-238268, V-238271, V-238277, V-238278, V-238279, V-238280, V-238281, V-238282, V-238283, V-238284, V-238285, V-238286, V-238287, V-238288, V-238289, V-238290, V-238291, V-238292, V-238293, V-238294, V-238295, V-238297, V-238299, V-238300, V-238301, V-238302, V-238303, V-238304, V-238309, V-238310, V-238315, V-238316, V-238317, V-238318, V-238319, V-238320, V-238324, V-238325, V-238329, V-238330, V-238332, V-238333, V-238334, V-238335, V-238337, V-238338, V-238339, V-238340, V-238341, V-238342, V-238343, V-238344, V-238345, V-238346, V-238347, V-238348, V-238349, V-238350, V-238351, V-238352, V-238353, V-238356, V-238358, V-238359, V-238360, V-238369, V-238370, V-238376, V-238377, V-238378, V-251505, und V-255912

STIG-Build-Linux-High-Version 2023.3.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht anwendbar ist, überspringt die Hardening-Komponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Organisationsspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Hardening-Komponente anwendet, z. B. wenn Administratoren Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die STIG-Build-Linux-High-Hardening-Komponenten enthalten alle aufgelisteten STIG-Einstellungen, die für STIG-Build-Linux-Low- und STIG-Build-Linux-Medium-Hardening-Komponenten AWSTOE gelten, zusätzlich zu den aufgelisteten STIG-Einstellungen, die speziell für Schwachstellen der Kategorie I gelten.

RHEL 7 STIG Version 3, Version 12

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) für diese Linux-Distribution anwendet, sowie:

- RHEL 7/CentOS 7

  V-204425, V-204442, V-204443, V-204447, V-204448, V-204455, V-204502, V-204620 und V-204621
- AL2:

  V-204425, V-204442, V-204443, V-204447, V-204448, V-204455, V-204502, V-204620 und V-204621

RHEL 8 STIG Version 1 Version 11

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) für diese Linux-Distribution anwendet, sowie:

- RHEL 8/CentOS 8/AL 2023

  V-230264, V-230265, V-230487, V-230492, V-230529, V-230531, V-230533 und V-230558

Ubuntu 18.04 STIG Version 2 Version 11

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) für diese Linux-Distribution anwendet, sowie:

V-219157, V-219158, V-219177, V-219212, V-219308, V-219313, V-219314, V-219316, V-251506 und V-251507

Ubuntu 20.04 STIG Version 1 Version 9

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Schwachstellen der Kategorien II und III (Medium und Low) für diese Linux-Distribution anwendet, sowie:

V-238201, V-238215, V-238218, V-238219, V-238326, V-238327, V-238380, V-251503 und V-251504

STIG-Versionsverlaufsprotokoll für Linux

In diesem Abschnitt wird der Versionsverlauf der Linux-Komponente protokolliert. Um die Änderungen und veröffentlichten Versionen für ein Quartal anzuzeigen, wählen Sie den Titel aus, um die Informationen zu erweitern.

Änderungen am Q3 2023 – 10/04/2023:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des dritten Quartals 2023 wie folgt:

STIG-Build-Linux-Low-Version 2023.3.x

- RHEL 7 STIG Version 3 Version 12
- RHEL 8 STIG Version 1 Version 11
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 9

STIG-Build-Linux-Medium Version 2023.3.x

- RHEL 7 STIG Version 3 Version 12
- RHEL 8 STIG Version 1 Version 11
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 9

STIG-Build-Linux-High-Version 2023.3.x

- RHEL 7 STIG Version 3 Version 12
- RHEL 8 STIG Version 1 Version 11
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 9

Änderungen für Q2 2023 – 05/03/2023:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des zweiten Quartals 2023 wie folgt:

STIG-Build-Linux-Low Version 2023.2.x

- RHEL 7 STIG Version 3 Version 11
- RHEL 8 STIG Version 1 Version 10
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 8

STIG-Build-Linux-Medium Version 2023.2.x

- RHEL 7 STIG Version 3 Version 11
- RHEL 8 STIG Version 1 Version 10
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 8

STIG-Build-Linux-High Version 2023.2.x

- RHEL 7 STIG Version 3 Version 11
- RHEL 8 STIG Version 1 Version 10
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 8

Änderungen am Q1 2023 – 03/27/2023:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des ersten Quartals 2023 wie folgt:

STIG-Build-Linux-Low Version 2023.1.x

- RHEL 7 STIG Version 3 Version 10
- RHEL 8 STIG Version 1 Version 9
- Ubuntu 18.04 STIG Version 2 Version 10

- Ubuntu 20.04 STIG Version 1 Version 7

STIG-Build-Linux-Medium Version 2023.1.x

- RHEL 7 STIG Version 3 Version 10
- RHEL 8 STIG Version 1 Version 9
- Ubuntu 18.04 STIG Version 2 Version 10
- Ubuntu 20.04 STIG Version 1 Version 7

STIG-Build-Linux-High-Version 2023.1.x

- RHEL 7 STIG Version 3 Version 10
- RHEL 8 STIG Version 1 Version 9
- Ubuntu 18.04 STIG Version 2 Version 10
- Ubuntu 20.04 STIG Version 1 Version 7

Änderungen für Q4 2022 – 02/01/2023:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des vierten Quartals 2022 wie folgt:

STIG-Build-Linux-Low Version 2022.4.x

- RHEL 7 STIG Version 3 Version 9
- RHEL 8 STIG Version 1 Version 8
- Ubuntu 18.04 STIG Version 2 Version 9
- Ubuntu 20.04 STIG Version 1 Version 6

STIG-Build-Linux-Medium Version 2022.4.x

- RHEL 7 STIG Version 3 Version 9
- RHEL 8 STIG Version 1 Version 8
- Ubuntu 18.04 STIG Version 2 Version 9
- Ubuntu 20.04 STIG Version 1 Version 6

STIG-Build-Linux-High-Version 2022.4.x

- RHEL 7 STIG Version 3 Version 9
- RHEL 8 STIG Version 1 Version 8
- Ubuntu 18.04 STIG Version 2 Version 9
- Ubuntu 20.04 STIG Version 1 Version 6

Änderungen für Q3 2022 – 09/30/2022 (keine Änderungen):

Für die Veröffentlichung des dritten Quartals 2022 wurden keine Änderungen für die Linux-Komponente STIGS vorgenommen.

Änderungen für Q2 2022 – 08/02/2022:

Ubuntu-Unterstützung eingeführt, STIG-Versionen aktualisiert und STIGS für das zweite Quartal 2022 angewendet:

STIG-Build-Linux-Low Version 2022.2.x

- RHEL 7 STIG Version 3 Version 7
- RHEL 8 STIG Version 1 Version 6
- Ubuntu 18.04 STIG Version 2 Release 6 (neu)
- Ubuntu 20.04 STIG Version 1 Release 4 (neu)

STIG-Build-Linux-Medium Version 2022.2.x

- RHEL 7 STIG Version 3 Version 7
- RHEL 8 STIG Version 1 Version 6
- Ubuntu 18.04 STIG Version 2 Release 6 (neu)
- Ubuntu 20.04 STIG Version 1 Release 4 (neu)

STIG-Build-Linux-High Version 2022.2.x

- RHEL 7 STIG Version 3 Version 7
- RHEL 8 STIG Version 1 Version 6
- Ubuntu 18.04 STIG Version 2 Release 6 (neu)

- Ubuntu 20.04 STIG Version 1 Release 4 (neu)

Änderungen für Q1 2022 – 04/26/2022:

Faktorwechsel, um eine bessere Unterstützung für Container einzuschließen. Das vorherige AL2-Skript wurde mit RHEL 7 kombiniert. Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des ersten Quartals 2022 wie folgt:

STIG-Build-Linux-Low Version 3.6.x

- RHEL 7 STIG Version 3 Version 6
- RHEL 8 STIG Version 1 Version 5

STIG-Build-Linux-Medium Version 3.6.x

- RHEL 7 STIG Version 3 Version 6
- RHEL 8 STIG Version 1 Version 5

STIG-Build-Linux-High Version 3.6.x

- RHEL 7 STIG Version 3 Version 6
- RHEL 8 STIG Version 1 Version 5

Änderungen für Q4 2021 – 12/20/2021:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des vierten Quartals 2021 wie folgt:

STIG-Build-Linux-Low Version 3.5.x

- RHEL 7 STIG Version 3 Version 5
- RHEL 8 STIG Version 1 Version 4

STIG-Build-Linux-Medium Version 3.5.x

- RHEL 7 STIG Version 3 Version 5
- RHEL 8 STIG Version 1 Version 4

STIG-Build-Linux-High Version 3.5.x

- RHEL 7 STIG Version 3 Version 5
- RHEL 8 STIG Version 1 Version 4

Änderungen im Q3 2021 – 09/30/2021:

Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des dritten Quartals 2021 wie folgt:

STIG-Build-Linux-Low Version 3.4.x

- RHEL 7 STIG Version 3, Version 4
- RHEL 8 STIG Version 1 Version 3

STIG-Build-Linux-Medium Version 3.4.x

- RHEL 7 STIG Version 3 Version 4
- RHEL 8 STIG Version 1 Version 3

STIG-Build-Linux-High Version 3.4.x

- RHEL 7 STIG Version 3 Version 4
- RHEL 8 STIG Version 1 Version 3

SCAP-Compliance-Validator-Komponente

Das Security Content Automation Protocol (SCAP) ist eine Reihe von Standards, anhand derer IT-Experten Sicherheitslücken bei der Anwendung für die Compliance identifizieren können. Der SCAP Compliance Checker (SCC) ist ein SCAP-validiertes Scan-Tool, das vom Naval Information Warfare Center (NIWC) Atlantic veröffentlicht wurde. Weitere Informationen finden Sie unter Security Content Automation Protocol (SCAP) Compliance Checker (SCC) auf der Atlantic-Website von NIWC.

Die `scap-compliance-checker-linux` Komponenten AWSTOE `scap-compliance-checker-windows` und laden den SCC-Kabel herunter und installieren ihn auf den Build- und Test-Instances der Pipeline. Wenn der Drucker ausgeführt wird, führt er mithilfe von DISA SCAP Benchmarks

authentifizierte Konfigurationsscans durch und stellt einen Bericht bereit, der die folgenden Informationen enthält. schreibt die Informationen AWSTOE auch in Ihre Anwendungsprotokolle.

- STIG-Einstellungen, die auf die Instance angewendet werden.
- Ein Gesamt-Compliance-Score für die Instance.

Wir empfehlen, als letzten Schritt Ihres Erstellungsprozesses eine SCAP-Validierung durchzuführen, um sicherzustellen, dass Sie genaue Ergebnisse der Compliance-Validierung melden.

> **Note**
>
> Sie können die Berichte mit einem der STIG Viewing Tools überprüfen. Diese Tools sind online über den DoD Cyber Exchange verfügbar.

In den folgenden Abschnitten werden die Benchmarks beschrieben, die die SCAP-Validierungskomponenten enthalten.

scap-compliance-checker-linux -Version 2021.04.0

Die `scap-compliance-checker-linux` Komponente wird auf den Build- und Test-Instances der Image-Builder-Pipeline ausgeführt. AWSTOE protokolliert sowohl den Bericht als auch den Wert, den die SCC-Anwendung generiert.

Die Komponente führt die folgenden Workflow-Schritte aus:

1. Lädt die SCC-Anwendung herunter und installiert sie.
2. Importiert die Compliance-Benchmarks.
3. Führt die Validierung mit der SCC-Anwendung aus.
4. Speichert den Compliance-Bericht und die Bewertung lokal auf dem Desktop der Build-Instance.
5. Protokolliert den Compliance-Score aus dem lokalen Bericht in den AWSTOE Anwendungsprotokolldateien.

> **Note**
>
> AWSTOE unterstützt derzeit die SCAP-Compliance-Validierung für Windows Server 2012 R2, 2016 und 2019.

Die SCAP-Compliance-Checker-Komponente für Windows enthält die folgenden Benchmarks:

SCC-Version: 5.4.2

Benchmarks für Q4 2021:

- U_MS_DotNet_Framework_4-0_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_IE11_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_2012_and_2012_R2_MS_V3R2_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Defender_AV_V2R2_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2016_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2019_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Firewall_V2R1_STIG_SCAP_1-2_Benchmark
- U_CAN_Ubuntu_18-04_V2R4_STIG_SCAP_1-2_Benchmark
- U_RHEL_7_V3R5_STIG_SCAP_1-2_Benchmark
- U_RHEL_8_V1R3_STIG_SCAP_1-2_Benchmark

scap-compliance-checker-linux -Version 2021.04.0

Die `scap-compliance-checker-linux` Komponente wird auf den Build- und Test-Instances der Image-Builder-Pipeline ausgeführt. AWSTOE protokolliert sowohl den Bericht als auch den Wert, den die SCC-Anwendung generiert.

Die Komponente führt die folgenden Workflow-Schritte aus:

1. Lädt die SCC-Anwendung herunter und installiert sie.
2. Importiert die Compliance-Benchmarks.
3. Führt die Validierung mit der SCC-Anwendung aus.
4. Speichert den Compliance-Bericht und die Bewertung lokal am folgenden Speicherort auf der Build-Instance: `/opt/scc/SCCResults`.
5. Protokolliert den Compliance-Score aus dem lokalen Bericht in den AWSTOE Anwendungsprotokolldateien.

> **Note**
>
> AWSTOE unterstützt derzeit die SCAP-Compliance-Validierung für RHEL 7/8 und Ubuntu 18. Die SCC-Anwendung unterstützt derzeit die x86-Architektur für die Validierung.

Die SCAP-Compliance-Checker-Komponente für Linux enthält die folgenden Benchmarks:

SCC-Version: 5.4.2

Benchmarks für Q4 2021:

- U_CAN_Ubuntu_18-04_V2R4_STIG_SCAP_1-2_Benchmark
- U_RHEL_7_V3R5_STIG_SCAP_1-2_Benchmark
- U_RHEL_8_V1R3_STIG_SCAP_1-2_Benchmark
- U_MS_DotNet_Framework_4-0_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_IE11_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_2012_and_2012_R2_MS_V3R2_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Defender_AV_V2R2_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2016_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2019_V2R1_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Firewall_V2R1_STIG_SCAP_1-2_Benchmark

SCAP-Versionsverlauf

In der folgenden Tabelle werden wichtige Änderungen an der SCAP-Umgebung und den in diesem Dokument beschriebenen Einstellungen beschrieben.

Änderung	Beschreibung	Datum
SCAP-Komponenten hinzugefügt	Es wurden die folgenden SCAP-Komponenten eingeführt: • Erstellte scap-compliance-checker-linux Version 2021.04.0 (SCC-Version: 5.4.2) •	20. Dezember 2021

Änderung	Beschreibung	Datum
	Erstellte scap-compliance-checker-linux Version 2021.04.0 (SCC-Version: 5.4.2)	

AWSTOE -Befehlsreferenz

AWSTOE ist eine Anwendung zur Komponentenverwaltung, die in der ausgeführt wirdAWS CLI.

> **Note**
>
> Einige AWSTOE Aktionsmodule erfordern erhöhte Berechtigungen, um auf einem Linux-Server ausgeführt zu werden. Um erhöhte Berechtigungen zu verwenden, stellen Sie der Befehlssyntax voran oder führen Sie den sudo su Befehl einmal aussudo, wenn Sie sich anmelden, bevor Sie die unten verknüpften Befehle ausführen. Weitere Informationen zu AWSTOE Aktionsmodulen finden Sie unter Vom AWSTOE Komponentenmanager unterstützte Aktionsmodule.

run

Verwenden Sie den run Befehl , um die YAML-Dokumentskripts für ein oder mehrere Komponentendokumente auszuführen.

validieren

Führen Sie den validate Befehl aus, um die YAML-Dokumentsyntax für ein oder mehrere Komponentendokumente zu überprüfen.

Befehl awstoe run

Dieser Befehl führt die YAML-Komponentendokumentskripts in der Reihenfolge aus, in der sie in der durch den `--config` Parameter angegebenen Konfigurationsdatei oder in der Liste der durch den `--documents` Parameter angegebenen Komponentendokumente enthalten sind.

> **Note**
>
> Sie müssen genau einen der folgenden Parameter angeben, niemals beide:

--config
--Dokumente

Syntax

```
awstoe run [--config <file path>] [--cw-ignore-failures <?>]
      [--cw-log-group <?>] [--cw-log-region us-west-2] [--cw-log-stream <?>]
      [--document-s3-bucket-owner <owner>] [--documents <file path,file path,...>]
      [--execution-id <?>] [--log-directory <file path>]
      [--log-s3-bucket-name <name>] [--log-s3-bucket-owner <owner>]
      [--log-s3-key-prefix <?>] [--parameters name1=value1,name2=value2...]
      [--phases <phase name>] [--state-directory <directory path>] [--version <?>]
      [--help] [--trace]
```

Parameter und Optionen

Parameter

--config *./config-example.json*

Kurzform: -c *./config-example.json*

Die Konfigurationsdatei (bedingt). Dieser Parameter enthält den Dateispeicherort für die JSON-Datei, die Konfigurationseinstellungen für die Komponenten enthält, die dieser Befehl ausführt. Wenn Sie run Befehlseinstellungen in einer Konfigurationsdatei angeben, dürfen Sie den `--documents` Parameter nicht angeben. Weitere Informationen zur Eingabekonfiguration finden Sie unter Konfigurieren der Eingabe für den AWSTOE Run-Befehl.

Gültige Standorte sind:

- Ein lokaler Dateipfad (*./config-example.json*)
- Ein S3-URI (`s3://`*bucket/key*)

--cw-ignore-failures

Kurzform: N/A

Ignorieren Sie Protokollierungsfehler aus den CloudWatch Protokollen.

--cw-log-group

Kurzform: N/A

Der `LogGroup` Name für die CloudWatch Protokolle.

--cw-log-region

Kurzform: N/A

Die AWS Region, die für die CloudWatch Protokolle gilt.

--cw-log-stream

Kurzform: N/A

Der `LogStream` Name für die CloudWatch Protokolle, der anweist, AWSTOE wohin die `console.log` Datei gestreamt werden soll.

--document-s3-bucket-owner

Kurzform: N/A

Die Konto-ID des Bucket-Eigentümers für S3-URI-basierte Dokumente.

--Dokumente*`./doc-1.yaml,./doc-n.yaml`*

Kurzform: -d, *`./doc-1.yaml./doc-n`*

Die Komponentendokumente (bedingt). Dieser Parameter enthält eine durch Komma getrennte Liste von Dateispeicherorten für die auszuführenden YAML-Komponentendokumente. Wenn Sie YAML-Dokumente für den run Befehl mit dem `--documents` Parameter angeben, dürfen Sie den `--config` Parameter nicht angeben.

Gültige Standorte sind:

- lokale Dateipfade (*`./component-doc-example.yaml`*).
- S3-URIs (`s3://`*`bucket/key`*).
- Image-Builder-Komponenten-Build-Versions-ARNs (arn:aws:imagebuilder:us-west-*`2:123456789012`*: component/*`my-example-component`*/2021.12.02/1).

> **Note**
>
> Es gibt keine Leerzeichen zwischen Elementen in der Liste, sondern nur Kommas.

--execution-id

Kurzform: -i

Dies ist die eindeutige ID, die für die Ausführung des aktuellen run Befehls gilt. Diese ID ist in den Ausgabe- und Protokolldateinamen enthalten, um diese Dateien eindeutig zu identifizieren und sie mit der aktuellen Befehlsausführung zu verknüpfen. Wenn diese Einstellung weggelassen wird, AWSTOE generiert eine GUID.

--log-verzeichnis

Kurzform: -l

Das Zielverzeichnis, in dem alle Protokolldateien aus dieser Befehlsausführung AWSTOE speichert. Standardmäßig befindet sich dieses Verzeichnis innerhalb des folgenden übergeordneten Verzeichnisses: `TOE_<DATETIME>_<EXECUTIONID>`. Wenn Sie das Protokollverzeichnis nicht angeben, AWSTOE verwendet das aktuelle Arbeitsverzeichnis (`.`).

--log-s3-bucket-name

Kurzform: -b

Wenn Komponentenprotokolle in Amazon S3 gespeichert sind (empfohlen), AWSTOElädt die Komponentenanwendungsprotokolle in den S3-Bucket mit dem Namen in diesem Parameter hoch.

--log-s3-bucket-owner

Kurzform: N/A

Wenn Komponentenprotokolle in Amazon S3 gespeichert werden (empfohlen), ist dies die Besitzerkonto-ID für den Bucket, in dem die Protokolldateien AWSTOEschreibt.

--log-s3-key-prefix

Kurzform: -k

Wenn Komponentenprotokolle in Amazon S3 gespeichert werden (empfohlen), ist dies das S3-Objektschlüsselpräfix für den Protokollspeicherort im Bucket.

--parameters ***name1***=***value1***,***name2***=***value2***...

Kurzform: N/A

Parameter sind veränderbare Variablen, die im Komponentendokument definiert sind, mit Einstellungen, die die aufrufende Anwendung zur Laufzeit bereitstellen kann.

-Phasen

Kurzform: -p

Eine durch Komma getrennte Liste, die angibt, welche Phasen aus den YAML-Komponentendokumenten ausgeführt werden sollen. Wenn ein Komponentendokument zusätzliche Phasen enthält, werden diese nicht ausgeführt.

--state-directory

Kurzform: -s

Der Dateipfad, in dem Statusverfolgungsdateien gespeichert werden.

--version

Kurzform: -v

Gibt die Version der Komponentenanwendung an.

Optionen

-h

Kurzform: -h

Zeigt ein Hilfehandbuch für die Verwendung der Optionen der Anwendung zur Komponentenverwaltung an.

--trace

Kurzform: -t

Aktiviert die ausführliche Protokollierung in der -Konsole.

awstoe-Validierungsbefehl

Wenn Sie diesen Befehl ausführen, validiert er die YAML-Dokumentsyntax für jedes der durch den `--documents` Parameter angegebenen Komponentendokumente.

Syntax

```
awstoe validate [--document-s3-bucket-owner <owner>]
      --documents <file path,file path,...> [--help] [--trace]
```

Parameter und Optionen

Parameter

--document-s3-bucket-owner

Kurzform: N/A

Quellkonto-ID der bereitgestellten S3-URI-basierten Dokumente.

--Dokumente*./doc-1.yaml,./doc-n.yaml*

Kurzform: -d, *./doc-1.yaml./doc-n*

Die Komponentendokumente (erforderlich). Dieser Parameter enthält eine durch Komma getrennte Liste von Dateispeicherorten für die auszuführenden YAML-Komponentendokumente. Gültige Standorte sind:

- Lokale Dateipfade (*./component-doc-example.yaml*)
- S3-URIs (`s3://`*bucket/key*)
- ARNs der Build-Version der Image-Builder-Komponente (arn:aws:imagebuilder:us-west-*2:123456789012*: component/*my-example-component*/2021.12.02/1)

> **Note**
>
> Es gibt keine Leerzeichen zwischen Elementen in der Liste, sondern nur Kommas.

Optionen

-h

Kurzform: -h

Zeigt ein Hilfehandbuch für die Verwendung der Optionen der Anwendung zur Komponentenverwaltung an.

--trace

Kurzform: -t

Aktiviert die ausführliche Protokollierung in der -Konsole.

Verwalten von EC2 Image Builder-Ressourcen

Ressourcen sind die Bausteine, aus denen Image-Pipelines bestehen, sowie die Images, die diese Pipelines erzeugen. In diesem Kapitel werden das Erstellen, Verwalten und Freigeben von Image-Builder-Ressourcen, einschließlich Komponenten, Rezepten und Images, sowie Infrastrukturkonfigurations- und Verteilungseinstellungen behandelt.

> **Note**
>
> Um Sie bei der Verwaltung Ihrer Image-Builder-Ressourcen zu unterstützen, können Sie jeder Ressource Ihre eigenen Metadaten in Form von Tags zuweisen. Sie verwenden Tags, um Ihre AWS Ressourcen auf unterschiedliche Weise zu kategorisieren, z. B. nach Zweck, Eigentümer oder Umgebung. Dies ist nützlich, wenn Sie viele Ressourcen desselben Typs haben. Sie können eine bestimmte Ressource anhand der ihr zugewiesenen Tags leichter identifizieren.
> Weitere Informationen zum Markieren Ihrer Ressourcen mit Image-Builder-Befehlen in der AWS CLIfinden Sie im Markieren von Ressourcen Abschnitt dieses Handbuchs.

Inhalt

Verwalten von Komponenten mit Image Builder

Image Builder verwendet die AWS Task Orchestrator and Executor (AWSTOE)-Komponentenverwaltungsanwendung, um komplexe Workflows zu orchestrieren. Erstellen und testen Sie Komponenten, die mit der AWSTOE Anwendung funktionieren, auf YAML-Dokumenten, die die Skripte zum Anpassen oder Testen Ihres Images definieren. Für AMI-Images installiert Image Builder Komponenten und die AWSTOE Komponentenverwaltungsanwendung auf seinen Amazon EC2-Build- und Test-Instances. Bei Container-Images werden die Komponenten und die AWSTOE Komponentenverwaltungsanwendung innerhalb des laufenden Containers installiert.

Image Builder verwendet AWSTOE, um alle Aktivitäten auf der Instance durchzuführen. Es ist keine zusätzliche Einrichtung erforderlich, um mit zu interagieren, AWSTOE wenn Sie Image-Builder-Befehle ausführen oder die Image-Builder-Konsole verwenden.

> **Note**
>
> Wenn eine von Amazon verwaltete Komponente das Ende ihrer Support-Lebensdauer erreicht hat, wird sie nicht mehr gewartet. Ca. vier Wochen davor erhalten alle Konten, die die Komponente verwenden, eine Benachrichtigung und eine Liste der betroffenen Rezepte in ihrem Konto von ihrem AWS Health Dashboard. Weitere Informationen zu AWS Health finden Sie im AWS Health-Benutzerhandbuch.

Workflow-Phasen zum Erstellen eines neuen Images

Der Image-Builder-Workflow zum Erstellen neuer Images umfasst die folgenden zwei verschiedenen Phasen.

1. Build-Phase (Pre-Snapshot) – Während der Build-Phase nehmen Sie Änderungen an der Amazon EC2-Build-Instance vor, auf der Ihr Basis-Image ausgeführt wird, um die Basislinie für Ihr neues Image zu erstellen. Ihr Rezept kann beispielsweise Komponenten enthalten, die eine Anwendung installieren oder die Firewall-Einstellungen des Betriebssystems ändern.

   Die folgenden Komponentenphasen werden während der Build-Phase ausgeführt:

   - build
   - validieren

   Nach erfolgreichem Abschluss dieser Phase erstellt Image Builder einen Snapshot oder ein Container-Image, das es für die Testphase und darüber hinaus verwendet.

2. Testphase (nach dem Snapshot) – Während der Testphase gibt es einige Unterschiede zwischen Bildern, die AMIs und Container-Imageserstellen. Für AMI-Workflows startet Image Builder eine EC2-Instance aus dem Snapshot, den es als letzten Schritt der Build-Phase erstellt hat. Tests werden auf der neuen Instance ausgeführt, um Einstellungen zu validieren und sicherzustellen, dass die Instance wie erwartet funktioniert. Bei Container-Workflows werden die Tests auf derselben Instance ausgeführt, die für die Erstellung verwendet wurde.

   Die folgende Komponentenphase wird für jede Komponente ausgeführt, die während der Testphase im Rezept enthalten ist:

   - Test

   Diese Komponentenphase gilt sowohl für Build- als auch für Testkomponententypen. Nach erfolgreichem Abschluss dieser Phase kann Image Builder Ihr endgültiges Image aus dem Snapshot oder dem Container-Image erstellen und verteilen.

> **Note**
>
> Mit AWSTOE können Sie viele Phasen in einem Komponentendokument definieren. Image Builder verfügt jedoch über strenge Regeln darüber, welche Phasen und in welchen Phasen es sie ausführt. Damit eine Komponente während der Build-Phase ausgeführt werden kann, muss das Komponentendokument mindestens eine dieser Phasen definieren: `build` oder `validate`. Damit eine Komponente während der Testphase ausgeführt werden kann, muss das Komponentendokument die `-test`Phase und keine anderen Phasen definieren.
> Da Image Builder die Stufen unabhängig ausführt, können verkettete Verweise in Komponentendokumenten die Stufengrenzen nicht überschreiten. Sie können einen Wert aus einer Phase, die in der Build-Phase ausgeführt wird, nicht mit einer Phase verketten, die in der Testphase ausgeführt wird. Sie können jedoch Eingabeparameter für das beabsichtigte Ziel definieren und Werte über die Befehlszeile übergeben. Weitere Informationen zum Festlegen von Komponentenparametern in Ihren Image-Builder-Rezepten finden Sie unter Verwalten von AWSTOE Komponentenparametern mit EC2 Image Builder.

Zur Unterstützung bei der Fehlerbehebung auf Ihrer Build- oder Test-Instance AWSTOE erstellt einen Protokollordner, der das Eingabedokument und die Protokolldateien enthält, um zu verfolgen, was bei jeder Ausführung einer Komponente passiert. Wenn Sie einen Amazon S3-Bucket in Ihrer Pipeline-Konfiguration konfiguriert haben, werden die Protokolle auch dort geschrieben. Weitere

Informationen zu YAML-Dokumenten und Protokollausgaben finden Sie unter Verwenden von Komponentendokumenten in AWSTOE.

> **Tip**
>
> Wenn Sie viele Komponenten verfolgen müssen, hilft Ihnen das Markieren dabei, eine bestimmte Komponente oder Version anhand der zugewiesenen Tags zu identifizieren. Weitere Informationen zum Markieren Ihrer Ressourcen mit Image-Builder-Befehlen in der AWS CLIfinden Sie im Markieren von Ressourcen Abschnitt dieses Handbuchs.

In diesem Abschnitt wird beschrieben, wie Sie Komponenten mithilfe der Image-Builder-Konsole oder der Befehle in auflisten, anzeigen, erstellen und importierenAWS CLI.

Inhalt

Erstellen eines YAML-Komponentendokuments

Um eine Komponente zu erstellen, geben Sie ein YAML-Anwendungskomponentendokument an. Dies stellt die Phasen und Schritte dar, die Sie zum Erstellen der Komponente benötigen.

Die Beispiele in diesem Abschnitt erstellen eine Build-Komponente, die das `Update0S` Aktionsmodul in der AWSTOE Komponentenverwaltungsanwendung aufruft. Das Modul aktualisiert das Betriebssystem. Weitere Informationen zum `Update0S` Aktionsmodul finden Sie unter UpdateOS. Weitere Informationen zu den Phasen, Schritten und der Syntax für AWSTOE YAML-Anwendungskomponentendokumente finden Sie unter Verwenden von Dokumenten in AWSTOE.

> **Note**
>
> Image Builder bestimmt die Komponententypen im Pipeline-Workflow. Dieser Workflow entspricht der Build-Phase und der Testphase im Build-Prozess. Image Builder bestimmt den Komponententyp wie folgt:
>
> - Build – Dies ist der Standardkomponententyp. Alles, was nicht als Testkomponente klassifiziert ist, ist eine Build-Komponente. Diese Art von Komponente wird während der Build-Phase ausgeführt. Wenn für diese Build-Komponente eine `-test`Phase definiert ist, wird diese Phase während der Testphase ausgeführt.
> - Test – Um sich als Testkomponente zu qualifizieren, darf das Komponentendokument nur eine Phase mit dem Namen enthalten`test`. Für Tests, die sich auf Konfigurationen von Build-Komponenten beziehen, empfehlen wir, keine eigenständige Testkomponente zu verwenden. Verwenden Sie stattdessen die `-test`Phase in der zugehörigen Build-Komponente.
>
> Weitere Informationen darüber, wie Image Builder Phasen und Stufen zur Verwaltung des Komponenten-Workflows im Erstellungsprozess verwendet, finden Sie unter Verwalten von Komponenten mit Image Builder.

Um ein YAML-Anwendungskomponentendokument für eine Beispielanwendung zu erstellen, führen Sie die Schritte auf der Registerkarte aus, die Ihrem Image-Betriebssystem entspricht.

Linux

Erstellen einer YAML-Komponentendatei

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*`update-linux-os.yaml`*. Fügen Sie den folgenden Inhalt ein:

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
```

```
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-linux-os
description: Updates Linux with the latest security updates.
schemaVersion: 1
phases:
  - name: build
    steps:
    - name: UpdateOS
      action: UpdateOS
# Document End
```

> **Tip**
>
> Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihre YAML gut formatiert ist.

Windows

Erstellen einer YAML-Komponentendatei

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*update-windows-os.yaml*. Fügen Sie den folgenden Inhalt ein:

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
```

```
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-windows-os
description: Updates Windows with the latest security updates.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: UpdateOS
        action: UpdateOS
# Document End
```

> **Tip**
>
> Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihre YAML gut formatiert ist.

Verwalten von AWSTOE Komponentenparametern mit EC2 Image Builder

Sie können AWSTOE Komponenten, einschließlich der Erstellung und Einstellung von Komponentenparametern, direkt über die EC2-Image-Builder-Konsole oder mithilfe von AWS CLI Befehlen oder eines der Image-Builder-SDKs verwalten. In diesem Abschnitt behandeln wir das Erstellen und Verwenden von Parametern in Ihrer Komponente und das Festlegen von Komponentenparametern über die Image-Builder-Konsole und -AWS CLIBefehle.

> **Important**
>
> Komponentenparameter sind Klartextwerte und werden in protokolliertAWS CloudTrail. Wir empfehlen Ihnen, AWS Secrets Manager oder den AWS Systems Manager Parameter Store zu verwenden, um Ihre Secrets zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch.

Weitere Informationen zu AWS Systems Manager Parameter Store finden Sie unter AWS Systems Manager Parameter Store im AWS Systems Manager -Benutzerhandbuch.

Verwenden von Parametern in Ihrem YAML-Komponentendokument

Um eine Komponente zu erstellen, geben Sie ein YAML-Anwendungskomponentendokument an. Dies stellt die Phasen und Schritte dar, die Sie zum Erstellen der Komponente benötigen. Das Rezept, das auf die Komponente verweist, kann die Parameter festlegen, um die Werte zur Laufzeit anzupassen, wobei Standardwerte gelten, wenn der Parameter nicht auf einen bestimmten Wert festgelegt ist.

Erstellen eines Komponentendokuments mit Eingabeparametern

In diesem Abschnitt erfahren Sie, wie Sie Eingabeparameter in Ihrem YAML-Komponentendokument definieren und verwenden.

Um ein YAML-Anwendungskomponentendokument zu erstellen, das Parameter verwendet und Befehle in Ihren Build- oder Test-Instances von Image Builder ausführt, folgen Sie den Schritten, die Ihrem Image-Betriebssystem entsprechen:

Linux

Erstellen eines YAML-Komponentendokuments

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*hello-world-test.yaml*. Fügen Sie den folgenden Inhalt ein:

```
# Document Start
#
name: "HelloWorldTestingDocument-Linux"
description: "Hello world document to demonstrate parameters."
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
```

```
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

> **Tip**
>
> Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihre YAML gut formatiert ist.

Windows

Erstellen eines YAML-Komponentendokuments

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*`hello-world-test.yaml`*. Fügen Sie den folgenden Inhalt ein:

```
# Document Start
#
name: "HelloWorldTestingDocument-Windows"
description: "Hello world document to demonstrate parameters."
```

```
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

Tip

Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihre YAML gut formatiert ist.

Weitere Informationen zu den Phasen, Schritten und der Syntax für AWSTOE YAML-Anwendungskomponentendokumente finden Sie unter Verwenden von Dokumenten in AWSTOE. Weitere Informationen zu Parametern und ihren Anforderungen finden Sie im Parameter Abschnitt auf der Seite Definieren und Referenzieren von Variablen in AWSTOE .

Erstellen einer Komponente aus dem YAML-Komponentendokument

Unabhängig davon, welche Methode Sie zum Erstellen einer -AWSTOEKomponente verwenden, ist das YAML-Anwendungskomponentendokument immer als Grundlage erforderlich.

- Informationen zur Verwendung der Image-Builder-Konsole zum Erstellen einer Komponente direkt aus Ihrem YAML-Dokument finden Sie unter Erstellen einer Komponente mit der Image-Builder-Konsole.
- Informationen zur Verwendung von Image-Builder-Befehlen in der AWS CLI zum Erstellen Ihrer Komponente finden Sie unter Erstellen von AWSTOE Komponenten mit Image Builder mit der AWS CLI. Ersetzen Sie den Namen des YAML-Dokuments in diesen Beispielen durch den Namen Ihres Hello World YAML-Dokuments (*`hello-world-test.yaml`*).

Festlegen von Komponentenparametern in einem Image-Builder-Rezept (Konsole)

Das Festlegen von Komponentenparametern funktioniert für Image-Rezepte und Container-Rezepte gleich. Wenn Sie ein neues Rezept oder eine neue Version eines Rezepts erstellen, wählen Sie aus den Listen Build-Komponenten und Testkomponenten aus, welche Komponenten aufgenommen werden sollen. Die Komponentenlisten enthalten Komponenten, die für das Basisbetriebssystem gelten, das Sie für Ihr Image ausgewählt haben.

Nachdem Sie eine Komponente ausgewählt haben, wird sie im Abschnitt Ausgewählte Komponenten direkt unter den Komponentenlisten angezeigt. Konfigurationsoptionen werden für jede ausgewählte Komponente angezeigt. Wenn für Ihre Komponente Eingabeparameter definiert sind, werden sie als erweiterbarer Abschnitt mit dem Namen Eingabeparameter angezeigt.

Die folgenden Parametereinstellungen werden für jeden Parameter angezeigt, der für Ihre Komponente definiert ist:

- Parametername (nicht bearbeitbar) – Der Name des Parameters.
- Beschreibung (nicht bearbeitbar) – Die Parameterbeschreibung
- Typ (nicht bearbeitbar) – Der Datentyp für den Parameterwert.

- Wert – Der Wert für den Parameter. Wenn Sie diese Komponente zum ersten Mal in diesem Rezept verwenden und ein Standardwert für die Komponente definiert wurde, wird der Standardwert im Feld Wert mit ausgegrautem Text angezeigt. Wenn kein anderer Wert eingegeben wird, AWSTOE verwendet den Standardwert.

Auflisten und Anzeigen von Komponentendetails

In diesem Abschnitt wird beschrieben, wie Sie Informationen finden und Details zu den AWS Task Orchestrator and Executor (AWSTOE)-Komponenten anzeigen, die Sie in Ihren EC2-Image-Builder-Rezepten verwenden.

Komponentendetails

- AWSTOE Komponenten auflisten
- Auflisten von Komponenten-Build-Versionen (AWS CLI)
- Abrufen von Komponentendetails (AWS CLI)
- Abrufen von Komponentenrichtliniendetails (AWS CLI)

AWSTOE Komponenten auflisten

Sie können eine der folgenden Methoden verwenden, um AWSTOE Komponenten aufzulisten und zu filtern.

AWS Management Console

Gehen Sie folgendermaßen vorAWS Management Console, um eine Liste der Komponenten in der anzuzeigen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Komponenten aus. Standardmäßig zeigt Image Builder eine Liste der Komponenten an, die Ihrem Konto gehören.
3. Sie können optional nach Komponenteneigentümerschaft filtern. Um Komponenten zu sehen, die Sie nicht besitzen, aber auf die Sie Zugriff haben, erweitern Sie die Dropdown-Liste Besitzertyp und wählen Sie einen der Werte aus. Die Liste der Besitzertypen befindet sich in der Suchleiste neben dem Suchfeld. Sie können aus den folgenden Werten auswählen:

- Schnellstart (von Amazon verwaltet) – Öffentlich verfügbare Komponenten, die Amazon erstellt und verwaltet.
- In meinem Besitz – Komponenten, die Sie erstellt haben. Dies ist die Standardauswahl.
- Für mich freigegeben – Komponenten, die andere aus ihrem Konto erstellt und für Sie freigegeben haben.
- Von Drittanbietern verwaltete Komponenten – Komponenten, die einem Drittanbieter gehören, den Sie in abonniert habenAWS Marketplace.

AWS CLI

Das folgende Beispiel zeigt, wie Sie mit dem list-components Befehl eine Liste der AWSTOE Komponenten zurückgeben, die Ihrem Konto gehören.

```
aws imagebuilder list-components
```

Sie können optional nach Komponenteneigentümerschaft filtern. Das Besitzerattribut definiert, wer Eigentümer der Komponenten ist, die Sie auflisten möchten. Standardmäßig gibt diese Anforderung eine Liste der Komponenten zurück, die Ihrem Konto gehören. Um die Ergebnisse nach Komponentenbesitzer zu filtern, geben Sie einen der folgenden Werte mit dem `--owner` Parameter an, wenn Sie den list-components Befehl ausführen.

Werte des Komponentenbesitzers

- Selbst
- Amazon
- ThirdParty
- Freigegeben

Die folgenden Beispiele zeigen den -list-componentsBefehl mit dem `---owner`Parameter zum Filtern von Ergebnissen.

```
aws imagebuilder list-components --owner Self
{
    "requestId": "012a3456-b789-01cd-e234-fa5678b9012b",
    "componentVersionList": [
        {
```

```
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.0",
            "name": "sample-component01",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.1",
            "name": "sample-component01",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

```
aws imagebuilder list-components --owner Amazon
```

```
aws imagebuilder list-components --owner Shared
```

```
aws imagebuilder list-components --owner ThirdParty
```

Auflisten von Komponenten-Build-Versionen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie mit dem list-component-build-versions Befehl Komponenten-Build-Versionen auflisten, die eine bestimmte semantische Version haben. Weitere Informationen zur semantischen Versionsverwaltung für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

```
aws imagebuilder list-component-build-versions --component-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/example-component/1.0.1
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentSummaryList": [
```

```
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
            "name": "examplecomponent",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "description": "An example component that builds, validates and tests an
 image",
            "changeDescription": "Updated version.",
            "dateCreated": "2020-02-19T18:53:45.940Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

Abrufen von Komponentendetails (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den -get-componentBefehl verwenden, um Komponentendetails abzurufen, wenn Sie den Amazon-Ressourcennamen (ARN) der Komponente angeben.

```
aws imagebuilder get-component --component-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1/1
   {
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11112",
    "component": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
        "name": "examplecomponent",
        "version": "1.0.1",
        "type": "BUILD",
        "platform": "Linux",
        "owner": "123456789012",
        "data": "name: HelloWorldTestingDocument\ndescription: This is hello world
 testing document... etc.\"\n",
        "encrypted": true,
        "dateCreated": "2020-09-24T16:58:24.444Z",
        "tags": {}
    }
```

```
}
```

Abrufen von Komponentenrichtliniendetails (AWS CLI)

Das folgende Beispiel zeigt, wie Sie mit dem get-component-policy Befehl Details zu einer Komponentenrichtlinie abrufen, wenn Sie den ARN der Komponente angeben.

```
aws imagebuilder get-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1
```

Erstellen einer Komponente mit der Image-Builder-Konsole

Gehen Sie folgendermaßen vor, um eine AWSTOE Anwendungskomponente über die Image-Builder-Konsole zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Komponenten aus. Wählen Sie dann Komponente erstellen aus.
3. Geben Sie auf der Seite Komponente erstellen unter Komponentendetails Folgendes ein:
   a. Image-Betriebssystem (OS). Geben Sie das Betriebssystem an, mit dem die Komponente kompatibel ist.
   b. Komponentenkategorie . Wählen Sie aus der Dropdownliste den Typ der Build- oder Testkomponente aus, die Sie erstellen.
   c. Komponentenname . Geben Sie einen Namen für die Komponente ein.
   d. Komponentenversion . Geben Sie die Versionsnummer der Komponente ein.
   e. Beschreibung. Geben Sie eine optionale Beschreibung ein, um Ihnen bei der Identifizierung der Komponente zu helfen.
   f. Beschreibung der Änderung . Geben Sie eine optionale Beschreibung an, um die an dieser Version der Komponente vorgenommenen Änderungen zu verstehen.
4. Im Abschnitt Definitionsdokument ist die Standardoption Dokumentinhalt definieren . Das Komponentendokument definiert die Aktionen, die Image Builder auf den Build- und Test-Instances ausführt, um Ihr Image zu erstellen.

   Geben Sie im Feld Inhalt den Inhalt Ihres YAML-Komponentendokuments ein. Um mit einem Hello World-Beispiel für Linux zu beginnen, wählen Sie die Option Beispiel verwenden. Weitere

Informationen zum Erstellen eines YAML-Komponentendokuments oder zum Kopieren und Einfügen des UpdateOS-Beispiels von dieser Seite finden Sie unter Erstellen eines YAML-Komponentendokuments.

5. Nachdem Sie die Komponentendetails eingegeben haben, wählen Sie Komponente erstellen aus.

> **Note**
>
> Um Ihre neue Komponente beim Erstellen oder Aktualisieren eines Rezepts anzuzeigen, wenden Sie den Filter In meinem Besitz auf die Liste der Build- oder Testkomponenten an. Der Filter befindet sich oben in der Komponentenliste neben dem Suchfeld.

6. Um eine Komponente zu löschen, aktivieren Sie auf der Seite Komponenten das Kontrollkästchen neben der Komponente, die Sie löschen möchten. Wählen Sie in der Dropdownliste Aktionen die Option Komponente löschen aus.

Gehen Sie folgendermaßen vor, um eine neue Komponentenversion zu erstellen:

1. Je nachdem, wo Sie beginnen:
   - Auf der Seite Komponentenliste – Aktivieren Sie das Kontrollkästchen neben dem Komponentennamen und wählen Sie dann im Menü Aktionen die Option Neue Version erstellen aus.
   - Wählen Sie auf der Komponentendetailseite – Wählen Sie die Schaltfläche Neue Version erstellen in der oberen rechten Ecke der Überschrift.
2. Die Komponenteninformationen werden bereits mit den aktuellen Werten gefüllt, wenn die Seite Komponente erstellen angezeigt wird. Folgen Sie den Schritten zum Erstellen einer Komponente, um die Komponente zu aktualisieren. Dadurch wird sichergestellt, dass Sie eine eindeutige semantische Version in die Komponentenversion eingeben. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

Erstellen einer Komponente mit der AWS CLI

In diesem Abschnitt wird beschrieben, wie Sie mit Image-Builder-Befehlen AWS Task Orchestrator and Executor (AWSTOE)-Komponenten aus erstellenAWS Command Line Interface. Um eine

Komponente zu erstellen, geben Sie ein YAML-Anwendungskomponentendokument an. Dies stellt die Phasen und Schritte dar, die Sie zum Erstellen der Komponente benötigen. Informationen zum Erstellen eines neuen YAML-Komponentendokuments finden Sie unter Erstellen eines YAML-Komponentendokuments.

Erstellen von AWSTOE Komponenten mit Image Builder mit der AWS CLI

In diesem Abschnitt erfahren Sie, wie Sie Image-Builder-Befehle wie folgt in der einrichten und verwenden, AWS CLI um eine AWSTOE Anwendungskomponente zu erstellen.

- Laden Sie Ihr YAML-Komponentendokument in einen S3-Bucket hoch, auf den Sie über die Befehlszeile verweisen können.
- Erstellen Sie die AWSTOE Anwendungskomponente mit dem create-component Befehl .
- Listen Sie Komponentenversionen mit dem list-components Befehl und einem Namensfilter auf, um zu sehen, welche Versionen bereits vorhanden sind. Sie können die Ausgabe verwenden, um zu bestimmen, welche nächste Version für Updates verwendet werden soll.

Um eine AWSTOE Anwendungskomponente aus einem YAML-Eingabedokument zu erstellen, führen Sie die Schritte aus, die Ihrer Image-Betriebssystemplattform entsprechen.

Linux

Speichern Ihres Anwendungskomponentendokuments in Amazon S3

Sie können einen S3-Bucket als Repository für das Quelldokument Ihrer AWSTOE Anwendungskomponente verwenden. Gehen Sie folgendermaßen vor, um Ihr Komponentendokument zu speichern:

- Hochladen des Dokuments in Amazon S3

  Wenn Ihr Dokument kleiner als 64 KB ist, können Sie diesen Schritt überspringen. Dokumente mit einer Größe von 64 KB oder mehr müssen in Amazon S3 gespeichert werden.

  ```
  aws s3 cp update-linux-os.yaml s3://my-s3-bucket/my-path/update-linux-os.yaml
  ```

Erstellen einer Komponente aus dem YAML-Dokument

Um den create-component Befehl zu optimieren, den Sie in der verwendenAWS CLI, erstellen Sie eine JSON-Datei, die alle Komponentenparameter enthält, die Sie an den Befehl übergeben möchten. Geben Sie den Speicherort des *update-linux-os.yaml* Dokuments an, das Sie in den vorherigen Schritten erstellt haben. Das `uri` Schlüssel-Wert-Paar enthält die Dateireferenz.

> **Note**
>
> Die Namenskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Anforderungsparameter der Image-Builder-API-Aktion angegeben ist. Informationen zum Überprüfen der API-Befehlsanforderungsparameter finden Sie unter dem CreateComponent Befehl in der EC2-Image-Builder-API-Referenz . Informationen zur Angabe der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI -Befehlsreferenz angegeben sind.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*create-update-linux-os-component.json*. Fügen Sie den folgenden Inhalt ein:

   ```
   {
    "name": "update-linux-os",
    "semanticVersion": "1.1.2",
    "description": "An example component that updates the Linux operating system",
    "changeDescription": "Initial version.",
    "platform": "Linux",
    "uri": "s3://my-s3-bucket/my-path/update-linux-os.yaml",
    "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
   b123-456b-7f89-0123456f789c",
    "tags": {
     "MyTagKey-purpose": "security-updates"
    }
   }
   ```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Erstellen der Komponente

   Verwenden Sie den folgenden Befehl, um die Komponente zu erstellen, wobei Sie auf den Dateinamen für die JSON-Datei verweisen, die Sie im vorherigen Schritt erstellt haben:

```
aws imagebuilder create-component --cli-input-json file://create-update-linux-os-component.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Windows

Speichern Ihres Anwendungskomponentendokuments in Amazon S3

Sie können einen S3-Bucket als Repository für das Quelldokument Ihrer AWSTOE Anwendungskomponente verwenden. Gehen Sie folgendermaßen vor, um Ihr Komponentendokument zu speichern:

- Hochladen des Dokuments in Amazon S3

  Wenn Ihr Dokument kleiner als 64 KB ist, können Sie diesen Schritt überspringen. Dokumente mit einer Größe von 64 KB oder mehr müssen in Amazon S3 gespeichert werden.

  ```
  aws s3 cp update-windows-os.yaml s3://my-s3-bucket/my-path/update-windows-os.yaml
  ```

Erstellen einer Komponente aus dem YAML-Dokument

Um den create-component Befehl zu optimieren, den Sie in der verwendenAWS CLI, erstellen Sie eine JSON-Datei, die alle Komponentenparameter enthält, die Sie an den Befehl übergeben möchten. Geben Sie den Speicherort des *update-windows-os.yaml* Dokuments an, das Sie in den vorherigen Schritten erstellt haben. Das `uri` Schlüssel-Wert-Paar enthält die Dateireferenz.

> **Note**
>
> Die Namenskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Anforderungsparameter der Image-Builder-API-Aktion angegeben ist. Informationen zum Überprüfen der API-Befehlsanforderungsparameter finden Sie unter dem CreateComponent Befehl in der EC2 Image Builder-API-Referenz . Informationen zur Angabe der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI -Befehlsreferenz angegeben sind.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*create-update-windows-os-component.json*. Fügen Sie den folgenden Inhalt ein:

   ```
   {
    "name": "update-windows-os",
    "semanticVersion": "1.1.2",
    "description": "An example component that updates the Windows operating system.",
    "changeDescription": "Initial version.",
    "platform": "Windows",
   ```

```
 "uri": "s3://my-s3-bucket/my-path/update-windows-os.yaml",
 "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
b123-456b-7f89-0123456f789c",
 "tags": {
  "MyTagKey-purpose": "security-updates"
 }
}
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Erstellen der Komponente

   Verwenden Sie den folgenden Befehl, um die Komponente zu erstellen, wobei Sie auf den Dateinamen für die JSON-Datei verweisen, die Sie im vorherigen Schritt erstellt haben:

```
aws imagebuilder create-component --cli-input-json file://create-update-windows-
os-component.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

AWSTOE -Komponenten-Versioning für Updates (AWS CLI)

AWSTOE -Komponentennamen und -versionen werden nach dem Komponentenpräfix in den Amazon-Ressourcennamen (ARN) der Komponente eingebettet. Jede neue Version einer

Komponente hat ihren eigenen eindeutigen ARN. Die Schritte zum Erstellen einer neuen Version sind genau dieselben wie die Schritte zum Erstellen einer neuen Komponente, solange die semantische Version für diesen Komponentennamen eindeutig ist. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

Um sicherzustellen, dass Sie die nächste logische Version zuweisen, rufen Sie zunächst eine Liste der vorhandenen Versionen für die Komponente ab, die Sie ändern möchten. Verwenden Sie den list-components Befehl mit der AWS CLIund filtern Sie nach dem Namen.

In diesem Beispiel filtern Sie nach dem Namen der Komponente, die Sie in den vorherigen Linux-Beispielen erstellt haben. Um die von Ihnen erstellte Komponente aufzulisten, verwenden Sie den Wert des -nameParameters aus der JSON-Datei, die Sie im create-component Befehl verwendet haben.

```
aws imagebuilder list-components --filters name="name",values="update-linux-os"
{
    "requestId": "123a4567-b890-123c-45d6-ef789ab0cd1e",
    "componentVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-
linux-os/1.0.0",
            "name": "update-linux-os",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-
linux-os/1.0.1",
            "name": "update-linux-os",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

Anhand Ihrer Ergebnisse können Sie bestimmen, was die nächste Version sein sollte.

Importieren einer Komponente (AWS CLI)

In einigen Szenarien ist es möglicherweise einfacher, mit einem bereits vorhandenen Skript zu beginnen. Für dieses Szenario können Sie das folgende Beispiel verwenden.

In diesem Beispiel wird davon ausgegangen, dass Sie eine Datei mit dem Namen haben *import-component.json* (wie gezeigt). Beachten Sie, dass die Datei direkt auf ein PowerShell Skript namens verweistAdminConfig.ps1, das bereits in hochgeladen ist*my-s3-bucket*. Derzeit SHELL wird für die Komponente unterstütztformat.

```
{
"name": "MyImportedComponent",
"semanticVersion": "1.0.0",
"description": "An example of how to import a component",
"changeDescription": "First commit message.",
"format": "SHELL",
"platform": "Windows",
"type": "BUILD",
"uri": "s3://my-s3-bucket/AdminConfig.ps1",
"kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/60763706-
b131-418b-8f85-3420912f020c"
}
```

Führen Sie den folgenden Befehl aus, um die Komponente zu importieren.

```
aws imagebuilder import-component --cli-input-json file://import-component.json
```

Bereinigen von -Ressourcen

Um unerwartete Gebühren zu vermeiden, stellen Sie sicher, dass Sie Ressourcen und Pipelines bereinigen, die Sie aus den Beispielen in diesem Leitfaden erstellt haben. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unter Löschen von EC2 Image Builder-Ressourcen.

Rezepte verwalten

Ein EC2 Image Builder-Rezept definiert das Basis-Image, das als Ausgangspunkt zum Erstellen eines neuen Images verwendet werden soll, zusammen mit der Reihe von Komponenten, die Sie hinzufügen, um Ihr Image anzupassen und zu überprüfen, ob alles wie erwartet funktioniert. Image

Builder bietet automatische Versionsoptionen für jede Komponente. Die Anzahl der Komponenten, die Sie auf ein Rezept anwenden können, ist insgesamt auf 20 Komponenten begrenzt. Dazu gehören sowohl Build- als auch Testkomponenten.

Nachdem Sie ein Rezept erstellt haben, können Sie es nicht mehr ändern oder ersetzen. Um Komponenten zu aktualisieren, nachdem Sie ein Rezept erstellt haben, müssen Sie ein neues Rezept oder eine neue Rezeptversion erstellen. Sie können Tags immer auf Ihre vorhandenen Rezepte anwenden. Weitere Informationen zum Markieren Ihrer Ressourcen mit Image-Builder-Befehlen in der AWS CLIfinden Sie im Markieren von Ressourcen Abschnitt dieses Handbuchs.

> **Tip**
>
> Sie können von Amazon verwaltete Komponenten in Ihren Rezepten verwenden oder Ihre eigenen benutzerdefinierten Komponenten mit der AWS Task Orchestrator and Executor (AWSTOE)-Anwendung entwickeln. Um zu beginnen, sehen Sie sich Erste Schritte mit AWSTOE an.

In diesem Abschnitt wird beschrieben, wie Sie Rezepte auflisten, anzeigen und erstellen.

Inhalt

Auflisten und Anzeigen von Image-Rezeptdetails

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder-Image-Rezepten anzeigen können.

Details zu Image-Rezepten

- Abrufen von Image-Rezeptdetails (AWS CLI)
- Abrufen von Details zur Image-Rezeptrichtlinie (AWS CLI)

Auflisten von Image-Rezepten (Konsole)

Gehen Sie folgendermaßen vor, um eine Liste der Image-Rezepte anzuzeigen, die unter Ihrem Konto in der Image-Builder-Konsole erstellt wurden:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Rezepte aus. Dies zeigt eine Liste der Image-Rezepte, die unter Ihrem Konto erstellt wurden.
3. Um Details anzuzeigen oder eine neue Rezeptversion zu erstellen, wählen Sie den Link Rezeptname. Dadurch wird die Detailansicht für das Rezept geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Rezeptnamen aktivieren und dann Details anzeigen auswählen.

Auflisten von Image-Rezepten (AWS CLI)

Das folgende Beispiel zeigt, wie Sie alle Ihre Image-Rezepte mithilfe der auflistenAWS CLI.

```
aws imagebuilder list-image-recipes
```

Anzeigen von Image-Rezeptdetails (Konsole)

Um Details für ein bestimmtes Image-Rezept mithilfe der Image-Builder-Konsole anzuzeigen, wählen Sie das zu überprüfende Image-Rezept aus, indem Sie die unter beschriebenen Schritte ausführenAuflisten von Image-Rezepten (Konsole).

Auf der Seite mit den Rezeptdetails können Sie:

- Löschen Sie das Rezept. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unter Löschen von EC2 Image Builder-Ressourcen.
- Erstellen Sie eine neue Version.

- Erstellen Sie eine Pipeline aus dem Rezept. Nachdem Sie Pipeline aus diesem Rezept erstellen ausgewählt haben, werden Sie zum Pipeline-Assistenten weitergeleitet. Weitere Informationen zum Erstellen einer Image-Builder-Pipeline mit dem Pipeline-Assistenten finden Sie unter . Erstellen einer Image-Pipeline mit dem EC2 Image Builder-Konsolenassistenten

> **Note**
>
> Wenn Sie eine Pipeline aus einem vorhandenen Rezept erstellen, ist die Option zum Erstellen eines neuen Rezepts nicht verfügbar.

Abrufen von Image-Rezeptdetails (AWS CLI)

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um die Details eines Image-Rezepts abzurufen, indem Sie seinen Amazon-Ressourcennamen (ARN) angeben.

```
aws imagebuilder get-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2020.12.03
```

Abrufen von Details zur Image-Rezeptrichtlinie (AWS CLI)

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um die Details einer Image-Rezeptrichtlinie abzurufen, indem Sie ihren ARN angeben.

```
aws imagebuilder get-image-recipe-policy --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2020.12.03
```

Auflisten und Anzeigen von Container-Rezeptdetails

In diesem Abschnitt werden die Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2-Image-Builder-Container-Rezepten anzeigen können.

Details zu Container-Rezepten

- Auflisten von Container-Rezepten in der Konsole
- Auflisten von Container-Rezepten mit der AWS CLI
- Anzeigen von Container-Rezeptdetails in der Konsole
- Abrufen von Container-Rezeptdetails mit der AWS CLI
- Abrufen von Details zu Container-Rezeptrichtlinien mit der AWS CLI

Auflisten von Container-Rezepten in der Konsole

Gehen Sie folgendermaßen vor, um eine Liste der Container-Rezepte anzuzeigen, die unter Ihrem Konto in der Image-Builder-Konsole erstellt wurden:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Container-Rezepte aus. Dies zeigt eine Liste der Container-Rezepte, die unter Ihrem Konto erstellt werden.
3. Um Details anzuzeigen oder eine neue Rezeptversion zu erstellen, wählen Sie den Link Rezeptname. Dadurch wird die Detailansicht für das Rezept geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Rezeptnamen aktivieren und dann Details anzeigen auswählen.

Auflisten von Container-Rezepten mit der AWS CLI

Das folgende Beispiel zeigt, wie Sie alle Ihre Container-Rezepte mithilfe der auflistenAWS CLI.

```
aws imagebuilder list-container-recipes
```

Anzeigen von Container-Rezeptdetails in der Konsole

Um Details für ein bestimmtes Container-Rezept mit der Image-Builder-Konsole anzuzeigen, wählen Sie das zu überprüfende Container-Rezept aus und führen Sie die unter beschriebenen Schritte ausAuflisten von Container-Rezepten in der Konsole.

Auf der Seite mit den Rezeptdetails können Sie Folgendes tun:

- Löschen Sie das Rezept. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unter Löschen von EC2 Image Builder-Ressourcen.
- Erstellen Sie eine neue Version.
- Erstellen Sie eine Pipeline aus dem Rezept. Nachdem Sie Pipeline aus diesem Rezept erstellen ausgewählt haben, werden Sie zum Pipeline-Assistenten weitergeleitet. Weitere Informationen zum Erstellen einer Image-Builder-Pipeline mit dem Pipeline-Assistenten finden Sie unter . Erstellen einer Image-Pipeline mit dem EC2 Image Builder-Konsolenassistenten

> **Note**
>
> Wenn Sie eine Pipeline aus einem vorhandenen Rezept erstellen, ist die Option zum Erstellen eines neuen Rezepts nicht verfügbar.

Abrufen von Container-Rezeptdetails mit der AWS CLI

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um die Details eines Container-Rezepts abzurufen, indem Sie seinen ARN angeben.

```
aws imagebuilder get-container-recipe --container-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:container-recipe/my-example-recipe/2020.12.03
```

Abrufen von Details zu Container-Rezeptrichtlinien mit der AWS CLI

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um die Details einer Container-Rezeptrichtlinie abzurufen, indem Sie ihren ARN angeben.

```
aws imagebuilder get-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-example-
recipe/2020.12.03
```

Erstellen einer neuen Version eines Image-Rezepts

In diesem Abschnitt wird beschrieben, wie Sie eine neue Version eines Image-Rezepts erstellen.

Inhalt

Erstellen einer neuen Image-Rezeptversion (Konsole)

Wenn Sie eine neue Rezeptversion erstellen, ist dies praktisch genauso wie das Erstellen eines neuen Rezepts. Der Unterschied besteht darin, dass bestimmte Details in den meisten Fällen so vorausgewählt sind, dass sie dem Basisrezept entsprechen. In der folgenden Liste werden die

Unterschiede zwischen dem Erstellen eines neuen Rezepts und dem Erstellen einer neuen Version eines vorhandenen Rezepts beschrieben.

Details zum Basisrezept in der neuen Version

- Name – kann nicht bearbeitet werden.
- Version – Erforderlich. Dieses Basisdetail ist nicht mit der aktuellen Version oder einer Art von Sequenz vorausgefüllt. Geben Sie die Versionsnummer, die Sie erstellen möchten, im Format <major>.<minor>.<patch> ein. Wenn die Version bereits vorhanden ist, tritt ein Fehler auf.
- Die Option Image auswählen – Vorab ausgewählt, aber Sie können sie bearbeiten. Wenn Sie Ihre Auswahl für die Quelle Ihres Basis-Images ändern, verlieren Sie möglicherweise andere Details, die von der von Ihnen gewählten ursprünglichen Option abhängen.

  Um Details anzuzeigen, die mit Ihrer Basis-Image-Auswahl verknüpft sind, wählen Sie die Registerkarte aus, die Ihrer Auswahl entspricht.

  Managed image

  - Image Operating System (OS) – nicht bearbeitbar.
  - Image name – Vorab ausgewählt, basierend auf der Kombination der Basis-Image-Optionen, die Sie für das vorhandene Rezept getroffen haben. Wenn Sie jedoch die Option Image auswählen ändern, verlieren Sie den vorausgewählten Image-Namen .
  - Optionen für die automatische Versionierung – Entspricht nicht Ihrem Basisrezept. Diese Image-Option verwendet standardmäßig die Option Ausgewählte Betriebssystemversion verwenden.

    > **Important**
    >
    > Wenn Sie die semantische Versionsverwaltung verwenden, um Pipeline-Builds zu starten, stellen Sie sicher, dass Sie diesen Wert in Aktuelle verfügbare Betriebssystemversion verwenden ändern. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

  AWS Marketplace image

  - Abonnements – Diese Registerkarte sollte geöffnet sein und das abonnierte Image von AWS Marketplace sollte entsprechend Ihrem Basisrezept vorausgewählt sein. Wenn Sie das Image ändern, das Ihr Rezept als Basis-Image verwendet, verlieren Sie möglicherweise andere Details, die vom ausgewählten Original-Image abhängen.

Weitere Informationen zu -AWS MarketplaceProdukten finden Sie unter Kaufen von Produkten im AWS Marketplace -Käuferhandbuch.

Custom AMI

- AMI-ID – Erforderlich. Diese Einstellung ist jedoch nicht mit Ihrem ursprünglichen Eintrag vorausgefüllt. Sie müssen die AMI-ID für Ihr Basis-Image eingeben.

- Instance-Konfiguration – Die Einstellungen sind vorausgewählt, aber Sie können sie bearbeiten.
  - Systems-Manager-Agent – Sie können dieses Kontrollkästchen aktivieren oder deaktivieren, um die Installation des Systems-Manager-Agenten auf dem neuen Image zu steuern. Das Kontrollkästchen ist standardmäßig deaktiviert, um den Systems Manager-Agenten in Ihr neues Image aufzunehmen. Um den Systems-Manager-Agenten aus dem endgültigen Image zu entfernen, aktivieren Sie das Kontrollkästchen, damit der Agent nicht in Ihrem AMI enthalten ist.
  - Benutzerdaten – Sie können diesen Bereich verwenden, um Befehle oder ein auszuführendes Befehlsskript bereitzustellen, wenn Sie Ihre Build-Instance starten. Dieser Wert ersetzt jedoch alle Befehle, die Image Builder möglicherweise hinzugefügt hat, um sicherzustellen, dass Systems Manager installiert ist. Diese Befehle enthalten das Bereinigungsskript, das Image Builder normalerweise für Linux-Images ausführt, bevor das neue Image erstellt wird.

    > **Note**
    >
    > - Wenn Sie Benutzerdaten eingeben, stellen Sie sicher, dass der Systems-Manager-Agent auf Ihrem Basis-Image vorinstalliert ist oder dass Sie die Installation in Ihre Benutzerdaten aufnehmen.
    > - Stellen Sie für Linux-Images sicher, dass Bereinigungsschritte ausgeführt werden, indem Sie einen Befehl zum Erstellen einer leeren Datei mit dem Namen `perform_cleanup` in Ihr Benutzerdatenskript aufnehmen. Image Builder erkennt diese Datei und führt das Bereinigungsskript aus, bevor das neue Image erstellt wird. Weitere Informationen und ein Beispielskript finden Sie unter Bewährte Methoden für die Sicherheit in EC2 Image Builder.

- Arbeitsverzeichnis – vorausgewählt, aber Sie können es bearbeiten.
- Komponenten – Komponenten, die bereits im Rezept enthalten sind, werden im Abschnitt Ausgewählte Komponenten am Ende jeder der Komponentenlisten angezeigt (Erstellen und Testen). Sie können die ausgewählten Komponenten entfernen oder an Ihre Bedürfnisse anpassen.

CIS-Hardening-Komponenten folgen nicht den Standardregeln für die Komponentenreihenfolge in Image-Builder-Rezepten. Die CIS-Harding-Komponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests mit Ihrem Ausgabe-Image ausgeführt werden.

> **Note**
>
> Build- und Testkomponentenlisten zeigen verfügbare Komponenten basierend auf dem Komponentenbesitzertyp an. Um Komponenten für Ihr Rezept hinzuzufügen oder zu aktualisieren, wählen Sie den Besitzertyp für die gesuchte Komponente aus. Wenn Sie beispielsweise eine Komponente hinzufügen möchten, die einem Basis-Image zugeordnet ist, das Sie in abonniert habenAWS Marketplace, wählen Sie `Third party managed` aus der Liste der Besitzertypen neben der Suchleiste aus.

Sie können die folgenden Einstellungen für die von Ihnen ausgewählte Komponente konfigurieren:

- Versioning-Optionen – vorausgewählt, aber Sie können sie ändern. Wir empfehlen Ihnen, die Option Neueste verfügbare Komponentenversion verwenden zu wählen, um sicherzustellen, dass Ihre Image-Builds immer die neueste Version der Komponente aufnehmen. Wenn Sie eine bestimmte Komponentenversion in Ihrem Rezept verwenden müssen, können Sie Komponentenversion angeben auswählen und die Version in das daraufhin angezeigte Feld Komponentenversion eingeben.
- Eingabeparameter – Zeigt Eingabeparameter an, die die Komponente akzeptiert. Der Wert ist mit dem Wert aus der vorherigen Version des Rezepts vorausgefüllt. Wenn Sie diese Komponente zum ersten Mal in diesem Rezept verwenden und ein Standardwert für die Komponente definiert wurde, wird der Standardwert im Feld Wert mit ausgegrautem Text angezeigt. Wenn kein anderer Wert eingegeben wird, AWSTOE verwendet den Standardwert.

> **Important**
>
> Komponentenparameter sind Klartextwerte und werden in protokolliertAWS CloudTrail. Wir empfehlen Ihnen, AWS Secrets Manager oder den AWS Systems Manager Parameter Store zu verwenden, um Ihre Secrets zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch. Weitere Informationen zu AWS Systems Manager Parameter Store

finden Sie unter AWS Systems Manager Parameter Store im AWS Systems Manager - Benutzerhandbuch.

Um die Einstellungen für Versioning-Optionen oder Eingabeparameter zu erweitern, können Sie den Pfeil neben dem Namen der Einstellung auswählen. Um alle Einstellungen für alle ausgewählten Komponenten zu erweitern, können Sie die Option Alle erweitern deaktivieren und aktivieren.

- Speicher (Volumes) – sind vorausgefüllt. Der Gerätename des Root-Volumes, der Snapshot und die IOPS-Auswahl sind nicht bearbeitbar. Sie können jedoch alle verbleibenden Einstellungen ändern, z. B. die Größe . Sie können auch neue Volumes hinzufügen und neue oder vorhandene Volumes verschlüsseln.

  Um Volumes für die Images zu verschlüsseln, die Image Builder unter Ihrem Konto in der Quellregion (in der der Build ausgeführt wird) erstellt, müssen Sie die Speichervolume-Verschlüsselung im Image-Rezept verwenden. Die Verschlüsselung, die während der Verteilungsphase des Builds ausgeführt wird, gilt nur für Images, die an andere Konten oder Regionen verteilt werden.

  > **Note**
  >
  > Wenn Sie die Verschlüsselung für Ihre Volumes verwenden, müssen Sie den Schlüssel für jedes Volume separat auswählen, auch wenn der Schlüssel derselbe ist, der für das Stamm-Volume verwendet wird.

So erstellen Sie eine neue Image-Rezeptversion:

1. Wählen Sie oben auf der Seite mit den Rezeptdetails die Option Neue Version erstellen aus. Dadurch gelangen Sie zur Seite Image-Rezept erstellen.
2. Um die neue Version zu erstellen, nehmen Sie Ihre Änderungen vor und wählen Sie dann Image-Rezept erstellen aus.

Weitere Informationen zum Erstellen eines Image-Rezepts beim Erstellen einer Image-Pipeline finden Sie Schritt 2: Rezept auswählen unter im Abschnitt Erste Schritte dieses Handbuchs.

Erstellen eines Image-Rezepts mit der AWS CLI

Gehen Sie folgendermaßen vorAWS CLI, um ein Image-Rezept mit dem Image Builder-`create-image-recipe`Befehl in der zu erstellen:

Voraussetzungen

Bevor Sie die Image-Builder-Befehle in diesem Abschnitt ausführen, um ein Image-Rezept aus der zu erstellenAWS CLI, müssen Sie die Komponenten erstellen, die das Rezept verwendet. Das Beispiel für ein Image-Rezept im folgenden Schritt bezieht sich auf Beispielkomponenten, die im Erstellen einer Komponente mit der AWS CLI Abschnitt dieses Handbuchs erstellt wurden.

Nachdem Sie Ihre Komponenten erstellt haben oder vorhandene Komponenten verwendet haben, notieren Sie sich die ARNs, die Sie in das Rezept aufnehmen möchten.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Sie können die gesamte Eingabe für den create-image-recipe Befehl mit Inline-Befehlsparametern bereitstellen. Der resultierende Befehl kann jedoch ziemlich lang sein. Um den Befehl zu optimieren, können Sie stattdessen eine JSON-Datei bereitstellen, die alle Rezepteinstellungen enthält.

   > **Note**
   >
   > Die Namenskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Anforderungsparameter der Image-Builder-API-Aktion angegeben ist. Informationen zum Überprüfen der API-Befehlsanforderungsparameter finden Sie unter dem CreateImageRecipe Befehl in der EC2 Image Builder-API-Referenz . Informationen zur Angabe der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI -Befehlsreferenz angegeben sind.

   Im Folgenden finden Sie eine Zusammenfassung der Parameter, die diese Beispiele angeben:

   - Name (Zeichenfolge, erforderlich) – Der Name des Image-Rezepts.
   - description (Zeichenfolge) – Die Beschreibung des Image-Rezepts.
   - parentImage (Zeichenfolge, erforderlich) – Das Image, das das Image-Rezept als Grundlage für Ihr benutzerdefiniertes Image verwendet. Der Wert kann der Basis-Image-ARN oder eine AMI-ID sein.

> **Note**
>
> Das Linux-Beispiel verwendet ein Image Builder-AMI und das Windows-Beispiel verwendet einen ARN.

- semanticVersion (Zeichenfolge, erforderlich) – Die semantische Version des Image-Rezepts im folgenden Format mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben: <major>.<minor>.<patch>. Ein Wert könnte beispielsweise sein`1.0.0`. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.
- components (array, required) – Enthält ein Array von `ComponentConfiguration` Objekten. Es muss mindestens eine Build-Komponente angegeben werden:

  > **Note**
  >
  > Image Builder installiert Komponenten in der Reihenfolge, in der Sie sie im Rezept angegeben haben. CIS-Hardening-Komponenten werden jedoch immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests mit Ihrem Ausgabe-Image ausgeführt werden.

  - componentARN (Zeichenfolge, erforderlich) – Der Komponenten-ARN.

    > **Tip**
    >
    > Um eines der Beispiele zum Erstellen Ihres eigenen Image-Rezepts zu verwenden, müssen Sie die Beispiel-ARNs durch die ARNs für die Komponenten ersetzen, die Sie für Ihr Rezept verwenden.

  - Parameter (Array von Objekten) – Enthält ein Array von `ComponentParameter` Objekten.

    > **Important**
    >
    > Komponentenparameter sind Klartextwerte und werden in protokolliertAWS CloudTrail. Wir empfehlen Ihnen, AWS Secrets Manager oder den AWS Systems Manager Parameter Store zu verwenden, um Ihre Secrets zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager?

im AWS Secrets Manager -Benutzerhandbuch. Weitere Informationen zu AWS Systems Manager Parameter Store finden Sie unter AWS Systems Manager Parameter Store im AWS Systems Manager -Benutzerhandbuch.

- Name (Zeichenfolge, erforderlich) – Der Name des festzulegenden Komponentenparameters.
- value (Array von Zeichenfolgen, erforderlich) – Enthält ein Array von Zeichenfolgen, um den Wert für den benannten Komponentenparameter festzulegen. Wenn für die Komponente ein Standardwert definiert ist und kein anderer Wert angegeben wird, AWSTOE verwendet den Standardwert.

- additionalInstanceConfiguration (Objekt) – Geben Sie zusätzliche Einstellungen und Startskripts für Ihre Build-Instances an.
  - systemsManagerAgent (Objekt) – Enthält Einstellungen für den Systems-Manager-Agenten auf Ihrer Build-Instance.
    - uninstallAfterBuild (Boolean) – Steuert, ob der Systems-Manager-Agent aus Ihrem endgültigen Build-Image entfernt wird, bevor das neue AMI erstellt wird. Wenn diese Option auf gesetzt ist`true`, wird der Agent aus dem endgültigen Image entfernt. Wenn die Option auf gesetzt ist`false`, bleibt der Agent in , sodass er im neuen AMI enthalten ist. Der Standardwert ist `false`.

      > **Note**
      >
      > Wenn das `uninstallAfterBuild` Attribut nicht in der JSON-Datei enthalten ist und die folgenden Bedingungen erfüllt sind, entfernt Image Builder den Systems Manager-Agenten aus dem endgültigen Image, sodass er im AMI nicht verfügbar ist:
      >
      > - ist `userDataOverride` leer oder wurde in der JSON-Datei weggelassen.
      > - Image Builder hat den Systems Manager-Agent automatisch auf der Build-Instance für ein Betriebssystem installiert, auf dem der Agent nicht auf dem Basis-Image vorinstalliert war.

  - userDataOverride (Zeichenfolge) – Geben Sie Befehle oder ein Befehlsskript an, das beim Starten Ihrer Build-Instance ausgeführt werden soll.

> **Note**
>
> Die Benutzerdaten sind immer Basis-64-codiert.. Zum Beispiel sind die folgenden Befehle als `IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg==` codiert:
>
> ```
> #!/bin/bash
> mkdir -p /var/bb/
> touch /var
> ```
>
> Das Linux-Beispiel verwendet diesen codierten Wert.

Linux

Das Basis-Image (`parentImage`-Eigenschaft) im folgenden Beispiel ist ein AMI. Wenn Sie ein AMI verwenden, müssen Sie Zugriff auf das AMI haben, und das AMI muss sich in der Quellregion befinden (dieselbe Region, in der Image Builder den Befehl ausführt). Speichern Sie die Datei als `create-image-recipe.json`und verwenden Sie sie im create-image-recipe Befehl .

```
{
"name": "BB Ubuntu Image recipe",
"description": "Hello World image recipe for Linux.",
"parentImage": "ami-0a01b234c5de6fabc",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/bb$"
 }
],
"additionalInstanceConfiguration": {
 "systemsManagerAgent": {
   "uninstallAfterBuild": true
 },
 "userDataOverride": "IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg=="
}
}
```

Windows

Das folgende Beispiel bezieht sich auf die neueste Version des Windows Server 2016 English Full Base-Images. Der ARN in diesem Beispiel verweist auf das neueste Image in der SKU basierend auf den von Ihnen angegebenen semantischen Versionsfiltern: `arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x`.

```
{
"name": "MyBasicRecipe",
"description": "This example image recipe creates a Windows 2016 image.",
"parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-component/2019.12.02/1"
 },
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-imported-component/1.0.0/1"
 }
]
}
```

> **Note**
>
> Weitere Informationen zur semantischen Versionsverwaltung für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

2. Erstellen des Rezepts

Verwenden Sie den folgenden Befehl, um das Rezept zu erstellen. Geben Sie den Namen der JSON-Datei an, die Sie im vorherigen Schritt im `--cli-input-json` Parameter erstellt haben:

```
aws imagebuilder create-image-recipe --cli-input-json file://create-image-recipe.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Importieren einer VM als Basis-Image in der Konsole

In diesem Abschnitt konzentrieren wir uns darauf, wie Sie eine virtuelle Maschine (VM) als Basis-Image für Ihr Image-Rezept importieren. Wir behandeln hier keine anderen Schritte im Zusammenhang mit der Erstellung eines Rezepts oder einer Rezeptversion. Weitere Schritte zum Erstellen eines neuen Image-Rezepts mit dem Pipeline-Erstellungsassistenten in der Image-Builder-Konsole finden Sie unter Erstellen einer Image-Pipeline (AMI). Weitere Schritte zum Erstellen eines neuen Image-Rezepts oder einer neuen Rezeptversion finden Sie unter Erstellen einer neuen Version eines Image-Rezepts.

Um eine VM als Basis-Image für Ihr Image-Rezept in der Image-Builder-Konsole zu importieren, führen Sie diese Schritte zusammen mit allen anderen erforderlichen Schritten aus, um Ihr Rezept oder Ihre Rezeptversion zu erstellen.

1. Wählen Sie im Abschnitt Image auswählen für das Basis-Image die Option Basis-Image importieren aus.
2. Wählen Sie das Image-Betriebssystem (OS) und die Betriebssystemversion wie gewohnt aus.

VM-Importkonfiguration

Wenn Sie Ihre VM aus der Virtualisierungsumgebung exportieren, erstellt dieser Prozess einen Satz von einer oder mehreren Festplattencontainerdateien, die als Snapshots Ihrer VM-Umgebung, Einstellungen und Daten dienen. Sie können diese Dateien verwenden, um Ihre VM als Basis-Image für Ihr Image-Rezept zu importieren. Weitere Informationen zum Importieren von VMs in Image Builder finden Sie unter . Importieren und Exportieren von VM-Images

Gehen Sie wie folgt vor, um den Speicherort Ihrer Importquelle anzugeben:

Importquelle

Geben Sie im Abschnitt Datenträgercontainer 1 die Quelle für den ersten VM-Image-Datenträgercontainer oder -Snapshot an, der importiert werden soll.

1. Quelle – Dies kann entweder ein S3-Bucket oder ein EBS-Snapshot sein.
2. S3-Speicherort des Datenträgers auswählen – Geben Sie den Speicherort in Amazon S3 ein, an dem Ihre Datenträgerabbilder gespeichert sind. Um nach dem Speicherort zu suchen, wählen Sie S3 durchsuchen aus.
3. Um einen Datenträgercontainer hinzuzufügen, wählen Sie Datenträgercontainer hinzufügen aus.

IAM-Rolle

Um Ihrer VM-Importkonfiguration eine IAM-Rolle zuzuordnen, wählen Sie die Rolle aus der Dropdown-Liste IAM-Rolle aus oder wählen Sie Neue Rolle erstellen, um eine neue zu erstellen. Wenn Sie eine neue Rolle erstellen, wird die Konsolenseite für IAM-Rollen in einer separaten Registerkarte geöffnet.

Erweiterte Einstellungen – optional

Die folgenden Einstellungen sind optional. Mit diesen Einstellungen können Sie Verschlüsselung, Lizenzierung, Tags und mehr für das vom Import erstellte Basis-Image konfigurieren.

Allgemeines

1. Geben Sie einen eindeutigen Name nfür das Basis-Image an. Wenn Sie keinen Wert eingeben, erbt das Basis-Image den Rezeptnamen.
2. Geben Sie eine Version für das Basis-Image an. Verwenden Sie das folgende Format: *<major>.<minor>.<patch>*. Wenn Sie keinen Wert eingeben, erbt das Basis-Image die Rezeptversion.
3. Sie können auch eine Beschreibung für das Basis-Image eingeben.

Basis-Image-Architektur

Um die Architektur Ihrer VM-Importquelle anzugeben, wählen Sie einen Wert aus der Liste Architektur aus.

Verschlüsselung

Wenn Ihre VM-Festplattenabbilder verschlüsselt sind, müssen Sie einen Schlüssel angeben, der für den Importvorgang verwendet werden soll. Um einen AWS KMS key für den Import anzugeben, wählen Sie einen Wert aus der Liste Verschlüsselung (KMS-Schlüssel) aus. Die Liste enthält KMS-Schlüssel, auf die Ihr Konto in der aktuellen Region zugreifen kann.

Lizenzverwaltung

Wenn Sie eine VM importieren, erkennt der Importvorgang automatisch das VM-Betriebssystem und wendet die entsprechende Lizenz auf das Basis-Image an. Abhängig von Ihrer Betriebssystemplattform lauten die Lizenztypen wie folgt:

- Lizenz enthalten – Eine entsprechende AWS Lizenz für Ihre Plattform wird auf Ihr Basis-Image angewendet.
- Bring Your Own License (BYOL) – Behält die Lizenz von Ihrer VM bei, falls zutreffend.

Um Lizenzkonfigurationen, die mit erstellt wurden, AWS License Manager an Ihr Basis-Image anzuhängen, wählen Sie aus der Liste Name der Lizenzkonfiguration aus. Weitere Informationen zu License Manager finden Sie unter Arbeiten mit AWS License Manager

> **Note**
>
> - Lizenzkonfigurationen enthalten Lizenzregeln, die auf den Bedingungen Ihrer Unternehmensvereinbarungen basieren.
> - Linux unterstützt nur BYOL-Lizenzen.

Tags (Basis-Image)

Tags verwenden Schlüssel-Wert-Paare, um Ihrer Image-Builder-Ressource durchsuchbaren Text zuzuweisen. Um Tags für das importierte Basis-Image anzugeben, geben Sie Schlüssel-Wert-Paare mit den Feldern Schlüssel und Wert ein.

Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Klicken Sie zum Entfernen eines Tags auf Tag entfernen.

Erstellen einer neuen Version eines Container-Rezepts

In diesem Abschnitt erfahren Sie, wie Sie eine neue Version eines Container-Rezepts erstellen.

Inhalt

Erstellen einer neuen Container-Rezeptversion mit der Konsole

Das Erstellen einer neuen Version eines Container-Rezepts ist praktisch dasselbe wie das Erstellen eines neuen Rezepts. Der Unterschied besteht darin, dass bestimmte Details in den meisten Fällen so vorausgewählt sind, dass sie dem Basisrezept entsprechen. In der folgenden Liste werden die Unterschiede zwischen dem Erstellen eines neuen Rezepts und dem Erstellen einer neuen Version eines vorhandenen Rezepts beschrieben.

Rezeptdetails

- Name – nicht bearbeitbar.
- Version – Erforderlich. Dieses Detail ist nicht mit der aktuellen Version oder einer Art von Sequenz vorausgefüllt. Geben Sie die Versionsnummer, die Sie erstellen möchten, im Format major.minor.patch ein. Wenn die Version bereits vorhanden ist, wird ein Fehler angezeigt.

Basis-Image

- Image-Option auswählen – Vorab ausgewählt, aber bearbeitbar. Wenn Sie Ihre Auswahl für die Quelle Ihres Basis-Images ändern, verlieren Sie möglicherweise andere Details, die von der von Ihnen gewählten ursprünglichen Option abhängen.

  Um Details anzuzeigen, die mit Ihrer Basis-Image-Auswahl verknüpft sind, wählen Sie die Registerkarte aus, die Ihrer Auswahl entspricht.

  Managed images

  - Image Operating System (OS) – nicht bearbeitbar.
  - Image name – Vorab ausgewählt, basierend auf der Kombination der Basis-Image-Optionen, die Sie für das vorhandene Rezept getroffen haben. Wenn Sie jedoch die Option Image auswählen ändern, verlieren Sie den vorausgewählten Image-Namen .
  - Optionen für die automatische Versionierung – Entspricht nicht Ihrem Basisrezept. Die Optionen für die automatische Versionsverwaltung verwenden verwenden verwenden standardmäßig die Option Ausgewählte Betriebssystemversion.

> **Important**
>
> Wenn Sie die semantische Versionsverwaltung verwenden, um Pipeline-Builds zu starten, stellen Sie sicher, dass Sie diesen Wert in Aktuelle verfügbare Betriebssystemversion verwenden ändern. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

ECR image

- Image Operating System (OS) – Vorab ausgewählt, aber bearbeitbar.
- Betriebssystemversion – Vorab ausgewählt, aber bearbeitbar.
- ECR-Image-ID – Vorausgefüllt, aber bearbeitbar.

Docker Hub image

- Image Operating System (OS) – kann nicht bearbeitet werden.
- Betriebssystemversion – Vorab ausgewählt, aber bearbeitbar.
- Docker-Image-ID – Vorausgefüllt, aber bearbeitbar.

Instance-Konfiguration

- AMI-ID – Vorausgefüllt, aber bearbeitbar.
- Speicher (Volumes)

  EBS-Volume 1 (AMI-Stamm) – Vorgefüllt. Sie können den Gerätenamen , Snapshot oder die IOPS-Auswahl des Root-Volumes nicht bearbeiten. Sie können jedoch alle verbleibenden Einstellungen ändern, z. B. die Größe . Sie können auch neue Volumes hinzufügen.

> **Note**
>
> Wenn Sie ein Basis-AMI angegeben haben, das von einem anderen Konto für Sie freigegeben wurde, müssen die Snapshots für alle angegebenen sekundären Volumes auch für Ihr Konto freigegeben werden.

Arbeitsverzeichnis

- Arbeitsverzeichnispfad – Vorgefüllt, aber bearbeitbar.

Komponenten

- Komponenten – Komponenten, die bereits im Rezept enthalten sind, werden im Abschnitt Ausgewählte Komponenten am Ende jeder der Komponentenlisten angezeigt (Erstellen und Testen). Sie können die ausgewählten Komponenten entfernen oder an Ihre Bedürfnisse anpassen.

  CIS-Hardening-Komponenten folgen nicht den Standardregeln für die Komponentenreihenfolge in Image-Builder-Rezepten. Die CIS-Harding-Komponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests mit Ihrem Ausgabe-Image ausgeführt werden.

  > **Note**
  >
  > Build- und Testkomponentenlisten zeigen verfügbare Komponenten basierend auf dem Komponentenbesitzertyp an. Um Komponenten für Ihr Rezept hinzuzufügen oder zu aktualisieren, wählen Sie den Besitzertyp für die gesuchte Komponente aus. Wenn Sie beispielsweise eine Komponente hinzufügen möchten, die einem Basis-Image zugeordnet ist, das Sie in abonniert habenAWS Marketplace, wählen Sie `Third party managed` aus der Liste der Besitzertypen neben der Suchleiste aus.

  Sie können die folgenden Einstellungen für die von Ihnen ausgewählte Komponente konfigurieren:

  - Versioning-Optionen – vorausgewählt, aber Sie können sie ändern. Wir empfehlen Ihnen, die Option Neueste verfügbare Komponentenversion verwenden zu wählen, um sicherzustellen, dass Ihre Image-Builds immer die neueste Version der Komponente aufnehmen. Wenn Sie eine bestimmte Komponentenversion in Ihrem Rezept verwenden müssen, können Sie Komponentenversion angeben auswählen und die Version in das daraufhin angezeigte Feld Komponentenversion eingeben.
  - Eingabeparameter – Zeigt Eingabeparameter an, die die Komponente akzeptiert. Der Wert ist mit dem Wert aus der vorherigen Version des Rezepts vorausgefüllt. Wenn Sie diese Komponente zum ersten Mal in diesem Rezept verwenden und ein Standardwert für die Komponente definiert wurde, wird der Standardwert im Feld Wert mit ausgegrautem Text angezeigt. Wenn kein anderer Wert eingegeben wird, AWSTOE verwendet den Standardwert.

> **Important**
>
> Komponentenparameter sind Klartextwerte und werden in protokolliertAWS CloudTrail. Wir empfehlen Ihnen, AWS Secrets Manager oder den AWS Systems Manager Parameter Store zu verwenden, um Ihre Secrets zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch. Weitere Informationen zu AWS Systems Manager Parameter Store finden Sie unter AWS Systems Manager Parameter Store im AWS Systems Manager - Benutzerhandbuch.

Um die Einstellungen für Versioning-Optionen oder Eingabeparameter zu erweitern, können Sie den Pfeil neben dem Namen der Einstellung auswählen. Um alle Einstellungen für alle ausgewählten Komponenten zu erweitern, können Sie die Option Alle erweitern deaktivieren und aktivieren.

Ziel-Repository

- Name des Ziel-Repositorys – Das Amazon-ECR-Repository, in dem Ihr Ausgabe-Image gespeichert wird, wenn in der Verteilungskonfiguration Ihrer Pipeline kein anderes Repository für die Region angegeben ist, in der die Pipeline ausgeführt wird (Region 1).

So erstellen Sie eine neue Container-Rezeptversion:

1. Wählen Sie oben auf der Detailseite des Container-Rezepts die Option Neue Version erstellen aus. Sie werden zur Seite Rezept erstellen für Container-Rezepte weitergeleitet.
2. Um die neue Version zu erstellen, nehmen Sie Ihre Änderungen vor und wählen Sie dann Rezept erstellen aus.

Weitere Informationen zum Erstellen eines Container-Rezepts beim Erstellen einer Image-Pipeline finden Sie Schritt 2: Rezept auswählen unter im Abschnitt Erste Schritte dieses Handbuchs.

Erstellen eines Container-Rezepts mit der AWS CLI

Gehen Sie folgendermaßen vorAWS CLI, um ein Image-Builder-Container-Rezept mit dem `imagebuilder create-container-recipe` Befehl in der zu erstellen:

Voraussetzungen

Bevor Sie die Image-Builder-Befehle in diesem Abschnitt ausführen, um ein Container-Rezept mit der zu erstellenAWS CLI, müssen Sie die Komponenten erstellen, die das Rezept verwenden wird. Das Beispiel für ein Container-Rezept im folgenden Schritt bezieht sich auf Beispielkomponenten, die im Erstellen einer Komponente mit der AWS CLI Abschnitt dieses Handbuchs erstellt wurden.

Nachdem Sie Ihre Komponenten erstellt haben oder vorhandene Komponenten verwendet haben, notieren Sie sich die ARNs, die Sie in das Rezept aufnehmen möchten.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Sie können die gesamte Eingabe für den create-container-recipe Befehl mit Inline-Befehlsparametern bereitstellen. Der resultierende Befehl kann jedoch ziemlich lang sein. Um den Befehl zu optimieren, können Sie stattdessen eine JSON-Datei bereitstellen, die alle Einstellungen für das Container-Rezept enthält.

   > **Note**
   >
   > Die Namenskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Anforderungsparameter der Image-Builder-API-Aktion angegeben ist. Informationen zum Überprüfen der API-Befehlsanforderungsparameter finden Sie unter dem CreateContainerRecipe Befehl in der EC2 Image Builder-API-Referenz . Informationen zur Angabe der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI -Befehlsreferenz angegeben sind.

   Im Folgenden finden Sie eine Zusammenfassung der Parameter in diesem Beispiel:

   - components (Array von Objekten, erforderlich) – Enthält ein Array von `ComponentConfiguration` Objekten. Es muss mindestens eine Build-Komponente angegeben werden:

     > **Note**
     >
     > Image Builder installiert Komponenten in der Reihenfolge, in der Sie sie im Rezept angegeben haben. CIS-Hardening-Komponenten werden jedoch immer zuletzt

ausgeführt, um sicherzustellen, dass die Benchmark-Tests mit Ihrem Ausgabe-Image ausgeführt werden.

- componentARN (Zeichenfolge, erforderlich) – Der Komponenten-ARN.

  > **Tip**
  >
  > Um das Beispiel zum Erstellen Ihres eigenen Container-Rezepts zu verwenden, ersetzen Sie die Beispiel-ARNs durch die ARNs für die Komponenten, die Sie für Ihr Rezept verwenden. Dazu gehören die AWS-Region, der Name und die Versionsnummer für jeden .

- Parameter (Array von Objekten) – Enthält ein Array von `ComponentParameter` Objekten.

  > **Important**
  >
  > Komponentenparameter sind Klartextwerte und werden in protokolliertAWS CloudTrail. Wir empfehlen Ihnen, AWS Secrets Manager oder den AWS Systems Manager Parameter Store zu verwenden, um Ihre Secrets zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch. Weitere Informationen zu AWS Systems Manager Parameter Store finden Sie unter AWS Systems Manager Parameter Store im AWS Systems Manager -Benutzerhandbuch.

  - name (Zeichenfolge, erforderlich) – Der Name des festzulegenden Komponentenparameters.
  - value (Array von Zeichenfolgen, erforderlich) – Enthält ein Array von Zeichenfolgen, um den Wert für den benannten Komponentenparameter festzulegen. Wenn für die Komponente ein Standardwert definiert ist und kein anderer Wert angegeben wird, AWSTOE verwendet den Standardwert.

- containerType (Zeichenfolge, erforderlich) – Der Typ des zu erstellenden Containers. Gültige Werte sind `DOCKER`.
- dockerfileTemplateData (Zeichenfolge) – Die Dockerfile-Vorlage, die zum Erstellen Ihres Images verwendet wird, ausgedrückt als Inline-Daten-Blob.

- Name (Zeichenfolge, erforderlich) – Der Name des Container-Rezepts.
- description (Zeichenfolge) – Die Beschreibung des Container-Rezepts.
- parentImage (Zeichenfolge, erforderlich) – Image, das das Container-Rezept als Grundlage für Ihr benutzerdefiniertes Image verwendet. Der Wert kann der Basis-Image-ARN oder eine AMI-ID sein.
- platformOverride (Zeichenfolge) – Gibt die Betriebssystemplattform an, wenn Sie ein benutzerdefiniertes Basis-Image verwenden.
- semanticVersion (Zeichenfolge, erforderlich) – Die semantische Version des Container-Rezepts, die im folgenden Format angegeben ist, mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben: <major>.<minor>.<patch>. Ein Beispiel wäre `1.0.0`. Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.
- Tags (Zeichenfolgenzuordnung) – Tags, die an das Container-Rezept angehängt sind.
- instanceConfiguration (Objekt) – Eine Gruppe von Optionen, mit denen eine Instance zum Erstellen und Testen von Container-Images konfiguriert werden kann.
  - Image (Zeichenfolge) – Die AMI-ID, die als Basis-Image für eine Container-Build- und Test-Instance verwendet werden soll. Wenn Sie diesen Wert nicht angeben, verwendet Image Builder das entsprechende Amazon-ECS-optimierte AMI als Basis-Image.
  - blockDeviceMappings (Array von Objekten) – Definiert die Blockgeräte, die zum Erstellen einer Instance aus dem im image Parameter angegebenen Image-Builder-AMI angefügt werden sollen.
    - deviceName (Zeichenfolge) – Das Gerät, für das diese Zuordnungen gelten.
    - ebs (Objekt) – Wird zur Verwaltung der Amazon-EBS-spezifischen Konfiguration für dieses Mapping verwendet.
      - deleteOnTermination (Boolean) – Wird verwendet, um das Löschen bei Beendigung des zugehörigen Geräts zu konfigurieren.
      - verschlüsselt (boolean) – Wird verwendet, um die Geräteverschlüsselung zu konfigurieren.
      - volumeSize (Ganzzahl) – Wird verwendet, um die Volume-Größe des Geräts zu überschreiben.
      - volumeType (Zeichenfolge) – Wird verwendet, um den Volume-Typ des Geräts zu überschreiben.

- targetRepository (Objekt, erforderlich) – Das Ziel-Repository für das Container-Image, wenn in der Verteilungskonfiguration Ihrer Pipeline kein anderes Repository für die Region angegeben ist, in der die Pipeline ausgeführt wird (Region 1).
  - repositoryName (Zeichenfolge, erforderlich) – Der Name des Container-Repositorys, in dem das Ausgabe-Container-Image gespeichert ist. Diesem Namen wird der Speicherort des Repositorys vorangestellt.
  - service (Zeichenfolge, erforderlich) – Gibt den Service an, in dem dieses Image registriert wurde.
- workingDirectory (Zeichenfolge) – Das Arbeitsverzeichnis für die Verwendung während Build- und Test-Workflows.

```
{
   "components": [
      {
         "componentArn": "arn:aws:imagebuilder:us-east-1:123456789012:component/
helloworldal2/x.x.x"
      }
   ],
   "containerType": "DOCKER",
   "description": "My Linux Docker container image",
   "dockerfileTemplateData": "FROM
 {{{ imagebuilder:parentImage }}}\n{{{ imagebuilder:environments }}}\n{{{ imagebuilder:comp
   "name": "amazonlinux-container-recipe",
   "parentImage": "amazonlinux:latest",
   "platformOverride": "Linux",
   "semanticVersion": "1.0.2",
   "tags": {
      "sometag" : "Tag detail"
   },
   "instanceConfiguration": {
      "image": "ami-1234567890",
      "blockDeviceMappings": [
         {
            "deviceName": "/dev/xvda",
            "ebs": {
               "deleteOnTermination": true,
               "encrypted": false,
               "volumeSize": 8,
               "volumeType": "gp2"
             }
```

```
            }
         ]
      },
      "targetRepository": {
         "repositoryName": "myrepo",
         "service": "ECR"
      },
      "workingDirectory": "/tmp"
}
```

2. Erstellen des Rezepts

   Verwenden Sie den folgenden Befehl, um das Rezept zu erstellen. Geben Sie den Namen der JSON-Datei an, die Sie im vorherigen Schritt im `--cli-input-json` Parameter erstellt haben:

   ```
   aws imagebuilder create-container-recipe --cli-input-json file://create-container-recipe.json
   ```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Bereinigen von -Ressourcen

Um unerwartete Gebühren zu vermeiden, stellen Sie sicher, dass Sie Ressourcen und Pipelines bereinigen, die Sie aus den Beispielen in diesem Handbuch erstellt haben. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unter Löschen von EC2 Image Builder-Ressourcen.

Verwalten von EC2 Image Builder-Images

Nachdem Sie Image-Ressourcen für AMI- oder Container-Images mit Image Builder erstellt haben, können Sie sie mithilfe der Image-Builder-Konsole, über die Image-Builder-API oder mit imagebuilder Befehlen in der verwaltenAWS CLI.

> **Tip**
>
> Wenn Sie mehrere Ressourcen desselben Typs haben, hilft Ihnen das Markieren dabei, eine bestimmte Ressource anhand der ihr zugewiesenen Tags zu identifizieren. Weitere Informationen zum Markieren Ihrer Ressourcen mit Image-Builder-Befehlen in der AWS CLIfinden Sie im Markieren von Ressourcen Abschnitt dieses Handbuchs.

In diesem Abschnitt wird beschrieben, wie Sie Images auflisten, anzeigen und erstellen. Informationen zu Image-Workflows und deren Verwaltung finden Sie unter Verwalten von Build- und Test-Workflows für EC2-Image-Builder-Images.

Inhalt

Auflisten von Images und Build-Versionen

Auf der Seite Images in der Image-Builder-Konsole können Sie Listen aller Image-Builder-Image-Ressourcen sehen, die Sie besitzen, die für Sie freigegeben sind und auf die Sie Zugriff haben. Die Listenergebnisse enthalten einige wichtige Details zu diesen Ressourcen.

Sie können auch alle Images in Ihrem Konto sehen, für die Workflow-Aktionen ausstehen.

Inhalt

- Auflisten von Images, die auf Aktionen warten
- Auflisten von Image-Build-Versionen

Auflisten von Images

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen zu Ihren Bildern auflisten können.

Sie können eine der folgenden Methoden verwenden, um Image-Builder-Image-Ressourcen aufzulisten, auf die Sie Zugriff haben. Informationen zur -API-Aktion finden Sie unter ListImages in der EC2 Image Builder-API-Referenz. Die zugehörige SDK-Anforderung finden Sie unter dem Link Siehe auch auf derselben Seite.

Inhalt

- Auflisten von Images in der Konsole
- Auflisten von Images mit -AWS CLIBefehlen

Auflisten von Images in der Konsole

Gehen Sie folgendermaßen vor, um die Seite mit der Liste der Images in der Konsole zu öffnen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Images aus.

Die Seite Images in der -Konsole ist in Registerkarten unterteilt, je nach Image-Eigentümerschaft oder Workflow-Aktionen, die noch ausstehen. In diesem Abschnitt werden die ersten drei Registerkarten behandelt, auf denen Bilder angezeigt werden, die Sie besitzen oder auf die Sie Zugriff haben.

Registerkarte „Konsole“: In meinem Besitz

Auf der Registerkarte In meinem Besitz können Sie die folgenden Filter verwenden, um die Ergebnisse der Bildliste zu optimieren.

- Sie können in der Suchleiste nach dem Namen ganz oder teilweise suchen.
- Sie können Images basierend auf ihrer Betriebssystemplattform (Windows oder Linux) filtern.

- Sie können Bilder basierend auf der Art der Ausgabe filtern, die sie erzeugen (AMI- oder Container-Image).
- Sie können die Filterquelle verwenden, um Images zu finden, die mit dem VMIE aus einer virtuellen Maschine importiert wurden.

Nach den Filtersteuerelementen zeigt die Registerkarte In meinem Besitz eine Liste der Image-Builder-Images an, die Sie erstellt haben, mit den folgenden Details für die aufgelisteten Ressourcen:

Name/Version

Image-Builder-Image-Ressourcennamen beginnen mit dem Rezeptnamen und der Version, aus der sie erstellt wurden. Wählen Sie den Link aus, um alle zugehörigen Image-Build-Versionen anzuzeigen.

Typ

Der Typ des Ausgabe-Images, das Image Builder für diese Image-Ressource erstellt (ein AMI oder ein Container-Image).

Plattform

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Image-Quelle

Der Ursprung des Basis-Images, das Image Builder zum Erstellen dieser Image-Ressource verwendet hat. Dies wird hauptsächlich verwendet, um Ergebnisse für Bilder zu filtern, die von einer virtuellen Maschine (VMIE) importiert wurden.

Zeitpunkt der Erstellung

Das Datum und die Uhrzeit, zu der Image Builder die aktuelle Version der Image-Ressource erstellt hat.

ARN

Der Amazon-Ressourcenname (ARN) der aktuellen Version der Image-Ressource.

Registerkarte „Konsole“: Für mich freigegeben

Auf der Registerkarte Mit mir geteilt können Sie die folgenden Filter verwenden, um die Ergebnisse der Bildliste zu optimieren.

- Sie können in der Suchleiste nach dem Namen ganz oder teilweise suchen.
- Sie können Images basierend auf ihrer Betriebssystemplattform (Windows oder Linux) filtern.
- Sie können Bilder basierend auf der Art der Ausgabe filtern, die sie erzeugen (AMI- oder Container-Image).
- Sie können die Filterquelle verwenden, um Images zu finden, die mit dem VMIE aus einer virtuellen Maschine importiert wurden.

Nach den Filtersteuerelementen zeigt die Registerkarte Für mich freigegeben eine Liste der Image-Builder-Images an, die für Sie freigegeben wurden, mit den folgenden Details für die aufgelisteten Ressourcen:

Image-Name

Der Name der Image-Ressource, die für Sie freigegeben wurde. Um ein freigegebenes Image in einem Rezept zu verwenden, wählen Sie die Option Verwaltete Images auswählen aus und ändern den Image-Ursprung in Für mich freigegebene Images.

Typ

Der Typ des Ausgabe-Images, das Image Builder für diese Image-Ressource erstellt (ein AMI oder ein Container-Image).

Version

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Image-Quelle

Der Ursprung des Basis-Images, das Image Builder zum Erstellen dieser Image-Ressource verwendet hat, falls zutreffend. Dies wird hauptsächlich verwendet, um Ergebnisse für Bilder zu filtern, die von einer virtuellen Maschine (VMIE) importiert wurden.

Plattform

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Zeitpunkt der Erstellung

Das Datum und die Uhrzeit, zu der Image Builder die Version der Image-Ressource erstellt hat, die für Sie freigegeben wurde.

Eigentümer

Der Besitzer der freigegebenen Image-Ressource.

ARN

Der Amazon-Ressourcenname (ARN) der Image-Ressourcenversion, die für Sie freigegeben wurde.

Registerkarte „Konsole“: Von Amazon verwaltet

Auf der Registerkarte Von Amazon verwaltet können Sie die folgenden Filter verwenden, um die Ergebnisse der Bildliste zu optimieren.

- Sie können in der Suchleiste nach dem Namen ganz oder teilweise suchen.
- Sie können Images basierend auf ihrer Betriebssystemplattform (Windows oder Linux) filtern.
- Sie können Bilder basierend auf der Art der Ausgabe filtern, die sie erzeugen (AMI- oder Container-Image).
- Sie können die Filterquelle verwenden, um Images zu finden, die mit dem VMIE aus einer virtuellen Maschine importiert wurden.

Nach den Filtersteuerelementen zeigt die Registerkarte Verwaltet von Amazon eine Liste der von Amazon verwalteten Image-Builder-Images an, die Sie als Basis-Images für Ihre Rezepte verwenden können. Image Builder zeigt die folgenden Details für aufgelistete Ressourcen an:

Image-Name

Der Name des verwalteten Images. Wenn Sie ein Rezept erstellen, ist der Standardwert für Ihr Basis-Image Schnellstart (von Amazon verwaltet). Die Images, die auf dieser Registerkarte aufgeführt sind, füllen die Image-Namensliste aus, die der Betriebssystemplattform zugeordnet ist, die Sie beim Erstellen eines Rezepts für Ihr Basis-Image auswählen.

Typ

Der Typ des Ausgabe-Images, das Image Builder für diese Image-Ressource erstellt (ein AMI oder ein Container-Image).

Version

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Plattform

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Zeitpunkt der Erstellung

Das Datum und die Uhrzeit, zu der Image Builder die Version der Image-Ressource erstellt hat, die für Sie freigegeben wurde.

Eigentümer

Amazon besitzt die verwalteten Images.

ARN

Der Amazon-Ressourcenname (ARN) der Image-Ressourcenversion, die für Sie freigegeben wurde.

Auflisten von Images mit -AWS CLIBefehlen

Wenn Sie den list-images Befehl in der ausführenAWS CLI, können Sie eine Liste der Images abrufen, die Sie besitzen oder auf die Sie Zugriff haben.

Das folgende Befehlsbeispiel zeigt, wie Sie den list-images Befehl ohne Filter verwenden, um alle Image-Builder-Image-Ressourcen aufzulisten, die Sie besitzen.

Beispiel: Auflisten aller Images

```
aws imagebuilder list-images
```

Ausgabe:


```
{
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  },
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-win/1.0.1",
    "name": "image-recipe-win",
    "type": "AMI",
```

```
    "version": "1.0.1",
    "platform": "Windows",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

Wenn Sie den list-images Befehl ausführen, können Sie Filter anwenden, um die Ergebnisse zu optimieren, wie das folgende Beispiel zeigt. Weitere Informationen zum Filtern Ihrer Ergebnisse finden Sie im Befehl list-images in der AWS CLI -Befehlsreferenz.

Beispiel: Filtern nach Linux-Images

```
aws imagebuilder list-images --filters name="platform",values="Linux"
```

Ausgabe:

```
{
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

Auflisten von Images, die auf Aktionen warten

Wenn Sie die `WaitForAction` Schrittaktion in Ihrem Image-Workflow verwenden, wird der Workflow angehalten, bis Sie ihm ein Signal senden, um die Verarbeitung fortzusetzen oder den Workflow fehlschlagen zu lassen. Sie können diese Schrittaktion verwenden, wenn Sie über einen externen Prozess verfügen, der ausgeführt werden muss, bevor Sie fortfahren. Sie können dann die verwenden`SendWorkflowStepAction`, um ein Signal an den angehaltenen Schritt an `RESUME` oder zu senden`STOP`. Sie können Ihren Workflow auch über die Konsole beenden oder fortsetzen.

Die folgenden Registerkarten zeigen, wie Sie eine Liste aller Image-Ressourcen in Ihrem Konto mit Workflow-Schritten abrufen, die derzeit angehalten werden, um zu warten, bis ein Signal fortgesetzt oder gestoppt wird. Die Registerkarten decken die Schritte der Konsole und den AWS CLI Befehl ab.

Sie können auch die API oder ein SDK verwenden, um eine Liste der Workflow-Schritte abzurufen, die auf Aktion warten. Informationen zur -API-Aktion finden Sie unter ListWaitingWorkflowSteps in der API-Referenz zu EC2 Image Builder. Die zugehörige SDK-Anforderung finden Sie unter dem Link Siehe auch auf derselben Seite.

Console

Gehen Sie folgendermaßen vor, um in der Konsole zur Registerkarte Warten auf Aktion zu gelangen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Images aus. Dadurch wird die Seite Images list geöffnet.
3. Wählen Sie auf der Listenseite die Registerkarte Warten auf Aktion aus.
4. (optional) Um einen Schritt zu beenden oder fortzusetzen, aktivieren Sie das Kontrollkästchen neben dem Namen und wählen Sie dann Schritt beenden oder Schritt fortsetzen aus. Sie können mehr als ein Kontrollkästchen aktivieren, um dieselbe Aktion für alle ausgewählten Schritte auszuführen.

Ausstehende Workflow-Schrittdetails

Zu den Workflow-Details für den ausstehenden Schritt gehören die folgenden:

- Image name – Der Name der Image-Ressource, für die der Schritt aussteht. Sie können den Namenslink auswählen, um die Detailseite für dieses Bild anzuzeigen.
- Name des ausstehenden Schritts – Der Name des Workflow-Schritts, der auf eine Aktion wartet.
- Schrittausführungs-ID – Identifiziert die Laufzeit-Instance des Workflow-Schritts eindeutig. Sie können die verknüpfte ID auswählen, um Laufzeitdetails für den Schritt anzuzeigen.
- Schrittstart – Der Zeitstempel, zu dem die Laufzeit-Instance des Workflow-Schritts gestartet wurde.
- Workflow-ARN – Der Amazon-Ressourcenname (ARN) des Workflows mit dem ausstehenden Schritt.
- Aktionen – Die Schrittaktion, die sich in einem Wartestatus befindet.

AWS CLI

Wenn Sie den list-waiting-workflow-steps Befehl in der ausführenAWS CLI, erhalten Sie eine Liste aller Images in Ihrem Konto, deren Workflow-Schritte auf eine Aktion warten, bevor Sie den Image-Erstellungsprozess abschließen.

Das folgende Befehlsbeispiel zeigt, wie Sie mit dem list-waiting-workflow-steps Befehl alle Images in Ihrem Konto mit Workflow-Schritten auflisten, die auf eine Aktion warten.

Beispiel: Auflisten von Images in Ihrem Konto mit wartenden Workflow-Schritten

```
aws imagebuilder list-waiting-workflow-steps
```

Ausgabe:

Die Ausgabe für dieses Beispiel zeigt ein Image im Konto mit einem Schritt, der auf eine Aktion wartet.

```
{
    "steps": [
        {
            "imageBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:image/example-image/1.0.0/8",
            "name": "WaitForAction",
            "workflowExecutionId": "wf-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "stepExecutionId": "step-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "workflowBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:workflow/test/wait-for-action/1.0.0/1",
            "startTime": "2023-11-21T23:21:23.609Z",
            "action": "WaitForAction"
        }
    ]
}
```

Auflisten von Image-Build-Versionen

Auf der Seite Image-Build-Versionen der Image-Builder-Konsole finden Sie eine Liste der Build-Versionen und zusätzliche Details für eine Image-Ressource, die Sie besitzen. Sie können auch Befehle oder Aktionen mit der Image-Builder-API, SDKs oder verwenden, AWS CLI um Image-Build-Versionen aufzulisten.

Sie können eine der folgenden Methoden verwenden, um Image-Build-Versionen für Image-Ressourcen aufzulisten, die Sie besitzen. Informationen zur API-Aktion finden Sie unter ListImageBuildVersions in der API-Referenz zu EC2 Image Builder. Die zugehörige SDK-Anforderung finden Sie unter dem Link Siehe auch auf derselben Seite.

Console

Versionsdetails

Zu den Details auf der Seite Image-Build-Versionen in der Image-Builder-Konsole gehören die folgenden:

- Version – Die Build-Version der Image-Ressource. In der Image-Builder-Konsole verlinkt die Version auf eine Image-Detailseite.
- Typ – Der Typ der Ausgabe, die Image Builder beim Erstellen dieser Image-Ressource verteilt hat (ein AMI oder ein Container-Image).
- Erstellungsdatum – Das Datum und die Uhrzeit, zu der Image Builder die Image-Build-Version erstellt hat.
- Image-Status – Der aktuelle Status der Image-Build-Version. Der Status kann sich auf den Image-Build oder die Image-Disposition beziehen. Während des Build-Prozesses wird beispielsweise möglicherweise der Status `Building` oder angezeigt`Distributing`. Für die Disposition des Images wird möglicherweise der Status `Deprecated` oder angezeigt`Deleted`.
- Fehlergrund – Der Grund für den Image-Status. Die Image-Builder-Konsole zeigt nur den Grund an, wenn der Build fehlschlägt (Image-Status ist gleich `Failed`).
- Sicherheitserkenntnisse – Die aggregierten Image-Scanergebnisse für die referenzierte Image-Build-Version.
- ARN – Der Amazon-Ressourcenname (ARN) für die referenzierte Version der Image-Ressource.
- Protokollstream – Ein Link zum Protokollstream-Detail für die referenzierte Image-Build-Version.

Auflisten von Versionen

Führen Sie die folgenden Schritte aus, um Image-Build-Versionen in der Image-Builder-Konsole aufzulisten:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.

2. Wählen Sie im Navigationsbereich Images aus. Standardmäßig zeigt die Image-Liste die aktuelle Version der einzelnen Images an, die Sie besitzen.
3. Um eine Liste aller Versionen für ein Image anzuzeigen, wählen Sie den Link zur aktuellen Version. Der Link öffnet die Seite Image-Build-Versionen, auf der alle Build-Versionen für ein bestimmtes Image aufgeführt sind.

AWS CLI

Wenn Sie den list-image-build-versions Befehl in der ausführenAWS CLI, erhalten Sie eine vollständige Liste der Build-Versionen für die angegebene Image-Ressource. Sie müssen Eigentümer des Images sein, um diesen Befehl ausführen zu können.

Das folgende Befehlsbeispiel zeigt, wie Sie mit dem list-image-build-versions Befehl alle Build-Versionen für das angegebene Image auflisten.

Beispiel: Auflisten von Build-Versionen für ein bestimmtes Image

```
aws imagebuilder list-image-build-versions --image-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0
```

Ausgabe:

Die Ausgabe für dieses Beispiel enthält zwei Build-Versionen für das angegebene Image-Rezept.

```
{
  "requestId": "12f3e45d-67cb-8901-af23-45ed678c9b01",
  "imageSummaryList": [
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/2",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/2",
    "platform": "Linux",
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T01:04:40.609Z",
    "outputResources": {
```

```
      "amis": [
        {
        "region": "us-west-2",
        "image": "ami-012b3456789012c3d",
        "name": "image-recipe-name 2023-03-10T01-05-12.541Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    },
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/1",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/1",
    "platform": "Linux",
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T00:07:16.384Z",
    "outputResources": {
      "amis": [
        {
        "region": "us-west-2",
        "image": "ami-0d1e23456789f0a12",
        "name": "image-recipe-name 2023-03-10T00-07-18.146132Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    }
  ]
}
```

> **Note**
>
> Die Ausgabe des list-image-build-versions Befehls enthält derzeit keine Sicherheitserkenntnisse oder Protokollstreams.

Anzeigen von Bilddetails

Auf der Seite mit den Image-Details in der Image-Builder-Konsole können Sie Details zu einer bestimmten Image-Ressource anzeigen, die Sie besitzen. Sie können auch Befehle oder Aktionen mit der Image-Builder-API, SDKs oder verwenden, AWS CLI um Image-Details abzurufen.

Weitere Informationen zu Ressourcen, die ein anderes über eine AWS Resource Access Manager (AWS RAM)-Ressourcenfreigabe für Sie AWS-Konto freigegeben hat, finden Sie unter Zugriff auf für Sie freigegebene AWS Ressourcen im AWS RAM -Benutzerhandbuch.

Inhalt

Anzeigen von Image-Details in der Image-Builder-Konsole

Die Seite mit den Image-Details in der Image-Builder-Konsole enthält einen Übersichtsabschnitt mit zusätzlichen Informationen, die in Registerkarten gruppiert sind. Die Seitenüberschrift ist der Name und die Build-Version des Rezepts, das das Image erstellt hat.

Abschnitte und Registerkarten mit Konsolendetails

- Abschnitt „Zusammenfassung“
- Registerkarte Ausgaberessourcen
- Registerkarte Infrastrukturkonfiguration
- Registerkarte Verteilungseinstellungen
- Registerkarte Workflow
- Registerkarte „Sicherheitsergebnisse“
- Registerkarte „Tags“

Abschnitt „Zusammenfassung“

Der Übersichtsabschnitt erstreckt sich über die Breite der Seite und enthält die folgenden Details. Diese Details werden immer angezeigt.

Rezept

Der Rezeptname und die Version, die die Build-Version nicht enthält. Wenn die Build-Version beispielsweise ist`sample-linux-recipe | 1.0.1/2`, ist das Rezept `sample-linux-recipe | 1.0.1`und die Build-Version ist `2`.

Erstellungsdatum

Das Datum und die Uhrzeit, zu der Image Builder die Image-Build-Version erstellt hat.

Image-Status

Der aktuelle Status der Image-Build-Version. Der Status kann sich auf den Image-Build oder die Image-Disposition beziehen. Während des Build-Prozesses wird beispielsweise möglicherweise der Status `Building` oder angezeigt`Distributing`. Für die Disposition des Images wird möglicherweise der Status `Deprecated` oder angezeigt`Deleted`.

Grund für das Fehlschlagen

Der Grund für den Image-Status. Die Image-Builder-Konsole zeigt nur den Grund an, wenn der Build fehlschlägt (Image-Status ist gleich `Failed`).

Registerkarte Ausgaberessourcen

Auf der Registerkarte Ausgaberessourcen werden Ausgabe- und Verteilungsdetails für die Image-Ressource aufgeführt, die derzeit angezeigt wird. Die Informationen, die Image Builder anzeigt, hängen wie folgt von der Art des Rezepts ab, mit dem die Pipeline das Image erstellt hat.

Image-Rezept

- Region – Die Verteilungsregion für das Amazon Machine Image (AMI) der Ausgabe, das in der Spalte Image angegeben ist.
- Image – Die ID des AMI, das Image Builder an das Ziel verteilt hat. Diese ID ist mit der Seite Amazon Machine Images (AMIs) in der Amazon EC2-Konsole verknüpft.

> **Note**
>
> Image Builder erstellt das AMI, nachdem es die Ausgabe-Image-Ressource erstellt hat und bevor es das AMI an das Ziel verteilt.

- Name – Der Name des AMI, das Image Builder an das Ziel verteilt hat.
- Beschreibung – Die optionale Beschreibung aus dem Image-Rezept, mit dem die Pipeline die Ausgabebildressource erstellt hat.
- Konto – Das AWS-Konto, dem die aktuell angezeigte Image-Builder-Image-Ressource gehört.

Container-Rezept

Image Builder zeigt die folgenden Details für die Ausgabe an, die aus einem Container-Rezept erstellt wurde.

- Region – Die Verteilungsregion für das Container-Image, das in der Spalte Image-URI angegeben ist.
- Image URI – Der URI des Ausgabecontainer-Images, das Image Builder an das ECR-Repository in der Zielregion verteilt hat.

> **Note**
>
> Image Builder zeigt eine Zeile pro Ziel an. Das Ausgabe-Image hat immer mindestens einen Eintrag für die Verteilung an das Konto, das das Image erstellt hat. Weitere Ziele können Verteilungen über -RegionenAWS-Konten, oder umfassenAWS Organizations. Weitere Informationen finden Sie unter Verwalten von EC2 Image Builder-Verteilungseinstellungen.

Registerkarte Infrastrukturkonfiguration

Auf der Registerkarte Infrastrukturkonfiguration werden die Amazon EC2-Infrastruktureinstellungen angezeigt, die Image Builder zum Erstellen und Testen des aktuell angezeigten Images verwendet hat. Image Builder zeigt immer den Namen der Infrastrukturkonfigurationsressource (Konfigurationsname) und ihren Amazon-Ressourcennamen (ARN) an. Wenn Ihre Infrastrukturkonfiguration die Werte festlegt, können zusätzliche Infrastrukturdetails Folgendes enthalten:

- Instance-Typen
- Ein Instance-Profil
- Netzwerkinfrastruktur
- Sicherheitsgruppeneinstellungen
- Ein Amazon S3-Speicherort, an dem Image Builder Anwendungsprotokolle speichert
- Ein Amazon EC2-Schlüsselpaar zur Fehlerbehebung
- Ein Amazon SNS-Thema für Ereignisbenachrichtigungen

Weitere Informationen finden Sie unter Verwalten der Infrastrukturkonfiguration von EC2 Image Builder.

Registerkarte Verteilungseinstellungen

Auf der Registerkarte Verteilungseinstellungen werden Einstellungen angezeigt, die Image Builder zur Verteilung Ihrer Ausgabe-Images verwendet hat. Image Builder zeigt immer den Namen der Verteilungskonfigurationsressource (Konfigurationsname) und ihren Amazon-Ressourcennamen (ARN) an. Zusätzliche Verteilungsdetails hängen wie folgt von der Art des Rezepts ab, das die Image-Builder-Pipeline zum Erstellen des Images verwendet hat:

Image-Rezept

Wenn Ihre Verteilungskonfigurationsressource die Werte festlegt, können zusätzliche Verteilungsdetails Folgendes umfassen:

- Region – Die Verteilungsregion für das Amazon Machine Image (AMI) der Ausgabe.
- Ausgabe-AMI-Name – Der Name des AMI, das Image Builder an das Ziel verteilt hat.
- Verschlüsselung (KMS-Schlüssel) – Wenn konfiguriert, verwendet AWS KMS key Image Builder zum Verschlüsseln des Images für die Verteilung an die Zielregion.
- Zielkonten für die Verteilung – Wenn Sie die kontoübergreifende Verteilung konfiguriert haben, zeigt diese Spalte eine durch Komma getrennte Liste von AWS-Konten an, mit denen das Ausgabebild in der Zielregion geteilt werden soll.
- Prinzipale mit geteilter Berechtigung – Eine durch Komma getrennte Liste der AWSPrinzipale, die über die Berechtigung zum Starten Ihres Images verfügen, z. B. - AWS-Konten oder -Gruppen AWS Organizations oder Organisationseinheiten (OUs).OUs

> **Note**
>
> Wenn Sie anderen Prinzipalen die Berechtigung zum Starten Ihres Abbilds erteilen, sind Sie immer noch Eigentümer des Abbilds. stellt Ihrem Konto alle Instances in AWS Rechnung, die Amazon EC2 von Ihrem Abbild aus startet.

- Zielkonten für eine schnellere Startkonfiguration –
- Zugeordnete Lizenzkonfigurationen – Die License Manager-Lizenzkonfigurations-ARNs, die dem AMI in der angegebenen Region zugeordnet werden sollen.
- Konfiguration der Startvorlage –
- Festlegen der Standardversion der Startvorlage –

Container-Rezept

Container-Verteilungen enthalten immer die folgenden Details:

- Region – Die Verteilungsregion für das Container-Image, das in der Spalte Image-URI angegeben ist.
- Image-URI – Der URI des Ausgabecontainer-Images, das Image Builder an das Amazon-ECR-Repository in der Zielregion verteilt hat.

> **Note**
>
> Image Builder zeigt eine Zeile pro Ziel an. Das Ausgabe-Image hat immer mindestens einen Eintrag für die Verteilung an das Konto, das das Image erstellt hat. Weitere Ziele können Verteilungen über -RegionenAWS-Konten, oder umfassenAWS Organizations. Weitere Informationen finden Sie unter Verwalten von EC2 Image Builder-Verteilungseinstellungen.

Registerkarte Workflow

Workflows definieren die Reihenfolge der Schritte, die Image Builder ausführt, wenn es ein neues Image erstellt. Alle Images verfügen über Build- und Test-Workflows. Container verfügen über einen zusätzlichen Workflow für die Verteilung. Auf der Registerkarte Workflow werden die entsprechenden Workflows angezeigt, die Image Builder für Ihr Image ausgeführt hat.

Filtern von Workflow-Typen

Image Builder zeigt standardmäßig zunächst die Zusammenfassung des Build-Workflows und die Workflow-Schritte an. Der Workflow-Filter zeigt jedoch alle Workflows an, die für Ihr Image ausgeführt oder abgeschlossen werden. Um einen anderen Workflow anzuzeigen, wählen Sie wie folgt aus der Liste aus:

Image-Workflows (AMI-Ausgabe)

- `build-image`
- `test-image`

Container-Workflows (Container-Ausgabe)

- `build-container`
- `test-container`
- `distribute-container`

> **Note**
>
> Wenn der Workflow noch nicht gestartet wurde, wird er nicht in der Liste angezeigt. Wenn Ihr Image-Build beispielsweise soeben begonnen hat, `build-image` ist der einzige Workflow-Typ, der in der Liste angezeigt wird. Wenn der nächste Workflow beginnt, fügt `test-image` Image Builder ihn der Liste hinzu.

Nach dem Workflow-Filter zeigt der ausgewählte Workflow eine Laufzeitübersicht, die die folgenden Details für jeden Workflow-Typ enthält:

Workflow-Status

Der aktuelle Laufzeitstatus für diesen Workflow. Die Werte können Folgendes enthalten:

- Ausstehend
- Übersprungen
- In Ausführung
- Completed
- Fehlgeschlagen

- Rollback-in-progress
- Rollback abgeschlossen

Ausführungs-ID

Eine eindeutige Kennung, die Image Builder zuweist, um Laufzeitressourcen bei jeder Ausführung eines Workflows zu verfolgen.

Starten

Der Zeitstempel, zu dem die Laufzeit-Instance dieses Workflows gestartet wurde.

Ende

Der Zeitstempel, zu dem diese Laufzeit-Instance des Workflows abgeschlossen wurde.

Gesamtzahl der Schritte

Die Gesamtzahl der Schritte im Workflow. Dies sollte der Summe der Schrittanzahl für Schritte entsprechen, die erfolgreich waren, übersprungen wurden und fehlgeschlagen sind.

Schritte erfolgreich

Eine Laufzeitanzahl für die Anzahl der Schritte im Workflow, die erfolgreich ausgeführt wurden.

Schritte fehlgeschlagen

Eine Laufzeitanzahl für die Anzahl der Schritte im Workflow, die fehlgeschlagen sind.

Übersprungene Schritte

Eine Laufzeitanzahl für die Anzahl der Schritte im Workflow, die übersprungen wurden.

Die Details in der folgenden Liste melden den aktuellen Status für alle Schritte in dieser Laufzeit-Instance des Workflows. Image Builder zeigt die gleichen Details für alle Image-Typen an.

Schritt #

Eine Zahl, die die Reihenfolge darstellt, in der Image Builder die Workflow-Schritte ausführt.

Schritt-ID

Eine eindeutige Kennung für den Workflow-Schritt, die zur Laufzeit zugewiesen wird.

Schrittstatus

Der aktuelle Laufzeitstatus des angegebenen Workflow-Schritts.

Rollback-Status

Der aktuelle Rollback-Status, wenn diese Laufzeit-Instance des Workflows fehlgeschlagen ist.

Schrittname

Der Name des angegebenen Workflow-Schritts.

Starten

Der Zeitstempel, zu dem der angegebene Schritt für diese Laufzeit-Instance des Workflows gestartet wurde.

Ende

Der Zeitstempel, zu dem der angegebene Schritt für diese Laufzeit-Instance des Workflows abgeschlossen wurde.

Registerkarte „Sicherheitsergebnisse“

Wenn Sie das Scannen aktiviert haben, werden auf der Registerkarte Sicherheitsergebnisse die Ergebnisse von Common Vulnerabilities and Exposures (CVE) angezeigt. Amazon Inspector hat diese Ergebnisse auf der Test-Instance identifiziert, die Image Builder gestartet hat, um Ihr neues Image zu erstellen. Um sicherzustellen, dass Image Builder Ergebnisse für Ihr Image erfasst, müssen Sie das Scannen wie folgt konfigurieren:

1. Aktivieren Sie Amazon Inspector-Scans für Ihr Konto. Weitere Informationen finden Sie unter Erste Schritte mit Amazon Inspector im Amazon Inspector-Benutzerhandbuch.
2. Aktivieren Sie Sicherheitserkenntnisse für die Pipeline, die dieses Image erstellt. Wenn Sie Sicherheitserkenntnisse für Ihre Pipeline aktivieren, speichert Image Builder einen Snapshot der Erkenntnisse, bevor die Test-Instance beendet wird. Weitere Informationen finden Sie unter Konfigurieren von Sicherheitsscans für Image-Builder-Images in der AWS Management Console.

Die Registerkarte Sicherheitsergebnisse enthält die folgenden Details für jede Schwachstelle, die Amazon Inspector für Ihr Image identifiziert hat.

Schweregrad

Der Schweregrad der CVE-Erkenntnis. Werte sind wie folgt:

- Nicht verteilt

- Informativ
- Niedrig
- Medium
- Hoch
- Kritisch

Die ID des Ergebnisses

Die eindeutige Kennung für die CVE-Erkenntnis, die Amazon Inspector beim Scannen der Test-Instance für Ihr Image erkannt hat. Die ID ist mit der Seite Sicherheitserkenntnisse > Nach Schwachstelle verknüpft. Weitere Informationen finden Sie unter Verwalten von Sicherheitsergebnissen für Image-Builder-Images in der AWS Management Console.

Quelle

Die Quelle der Schwachstelleninformationen für die CVE-Erkenntnis.

Alter

Die Anzahl der Tage, seit das Ergebnis zum ersten Mal für Ihr Bild beobachtet wurde.

Inspector-Score

Der Wert, den Amazon Inspector für die CVE-Erkenntnis zugewiesen hat.

Registerkarte „Tags“

Auf der Registerkarte Tags werden alle Tags angezeigt, die Sie für Ihr Image definiert haben.

Abrufen von Image-Richtliniendetails (AWS CLI)

Das folgende Beispiel zeigt, wie Sie die Details einer Image-Richtlinie mit ihrem Amazon-Ressourcennamen (ARN) abrufen.

```
aws imagebuilder get-image-policy --image-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/example-image/2019.12.02
```

Erstellen von Images

In diesem Abschnitt erfahren Sie, wie Sie Image-Builder-Images erstellen und einen Build abbrechen, der gerade ausgeführt wird.

Inhalt

Erstellen eines Images

Es gibt verschiedene Möglichkeiten, ein neues Image Builder-Image zu erstellen. Sie können beispielsweise eine der folgenden Methoden verwenden, um ein Image mit der AWS Management Console oder zu erstellenAWS CLI. Sie können auch die CreateImage-API-Aktion verwenden. Die zugehörige SDK-Anfrage finden Sie unter dem Link Siehe auch für diesen Befehl in der EC2 Image Builder API-Referenz .

AWS Management Console

Um ein neues Image aus einer vorhandenen Pipeline zu erstellen, können Sie die Pipeline wie folgt manuell ausführen. Sie können den Pipeline-Assistenten auch verwenden, um ein neues Image von Grund auf neu zu erstellen. Siehe Erstellen einer Image-Pipeline (AMI) oder Erstellen einer Image-Pipeline (Docker), je nach Art des Bildes, das Sie erstellen möchten.

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Pipelines aus.
3. Aktivieren Sie das Kontrollkästchen neben dem Pipeline-Namen, den Sie ausführen möchten.
4. Um das Image zu erstellen, wählen Sie im Menü Aktionen die Option Pipeline ausführen aus. Dadurch wird die Pipeline gestartet.

Sie können auch einen Zeitplan für die Ausführung Ihrer Pipeline angeben oder Amazon EventBridge verwenden, um Ihre Pipeline basierend auf von Ihnen konfigurierten Regeln auszuführen.

AWS CLI

Bevor Sie den create-image Befehl in der ausführenAWS CLI, müssen Sie die folgenden Ressourcen erstellen, sofern sie noch nicht vorhanden sind:

Erforderliche -Ressourcen

- Rezept – Sie müssen genau ein Rezept für Ihr Bild wie folgt angeben:

Image-Rezept

Geben Sie den Amazon-Ressourcennamen (ARN) für Ihre Image-Rezeptressource mit dem `--image-recipe-arn` Parameter an.

Container-Rezept

Geben Sie den ARN für Ihre Container-Rezeptressource mit dem `--container-recipe-arn` Parameter an.

- Infrastrukturkonfiguration – Geben Sie den ARN für Ihre Infrastrukturkonfigurationsressource mit dem `--infrastructure-configuration-arn` Parameter an.

Sie können auch eine der folgenden Ressourcen angeben, die Ihr Image benötigt:

Optionale Ressourcen und Konfiguration

- Verteilungskonfiguration – Standardmäßig verteilt Image Builder die Ausgabe-Image-Ressource an Ihr Konto in der Region, in der Sie den create-image Befehl ausführen. Um zusätzliche Ziele oder Konfigurationen für Ihre Verteilung bereitzustellen, geben Sie den ARN für Ihre Verteilungskonfigurationsressource mit dem `--distribution-configuration-arn` Parameter an.
- Scannen von Bildern – Verwenden Sie den `--image-scanning-configuration` Parameter , um Snapshots für Amazon Inspector-Ergebnisse auf Ihrem Image oder Ihrer Container-Test-Instance zu konfigurieren. Für Container-Images geben Sie auch das ECR-Repository an, das Amazon Inspector für seine Scans verwendet.
- Bildtests – Um die Image-Builder-Testphase zu unterdrücken, verwenden Sie den `--image-tests-configuration` Parameter . Alternativ können Sie ein Timeout festlegen, wie lange es ausgeführt werden kann.
- Image-Tags – Verwenden Sie den Parameter `--tags`, um Ihrem Ausgabebild Tags hinzuzufügen.
- Image-Workflows – Wenn Sie keine Build- oder Test-Workflows angeben, erstellt Image Builder Ihr Image mit seinem Standard-Image-Workflow. Um Workflows anzugeben, die Sie erstellt haben, verwenden Sie den `---workflows`Parameter.

> **Note**
>
> Wenn Sie Image-Workflows angeben, müssen Sie auch den Namen oder ARN der IAM-Rolle angeben, die Image Builder zum Ausführen Ihrer Workflow-Aktionen im `--execution-role`Parameter verwendet.

Das folgende Beispiel zeigt, wie Sie ein Image mit dem AWS CLI Befehl create-image erstellen. Weitere Informationen finden Sie in der AWS CLI-Befehlsreferenz.

Beispiel: Erstellen eines Basis-Images mit Standardverteilung

```
aws imagebuilder create-image --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/simple-recipe-linux/1.0.0 --infrastructure-configuration-arn arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/simple-infra-config-linux
```

Ausgabe:

```
{
    "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
    "imageVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/simple-recipe-linux/1.0.0",
            "name": "simple-recipe-linux",
            ...
        }
    ]
}
```

Abbilderstellung abbrechen (AWS CLI)

Um einen laufenden Image-Build abzubrechen, verwenden Sie den cancel-image-creation Befehl wie folgt:

```
aws imagebuilder cancel-image-creation --image-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-recipe/2019.12.03/1
```

Importieren eines VM-Images

Image Builder lässt sich in die Amazon EC2 VM Import/Export-API integrieren, damit der Importvorgang im Hintergrund asynchron ausgeführt werden kann. Image Builder verweist auf die Aufgaben-ID aus dem VM-Import, um seinen Fortschritt zu verfolgen, und erstellt eine Image-Builder-Image-Ressource als Ausgabe. Auf diese Weise können Sie auf die Image-Builder-Image-Ressource in Ihren Rezepten verweisen, bevor der VM-Import abgeschlossen ist.

Importieren einer VM (Konsole)

Gehen Sie folgendermaßen vor, um eine VM mit der Image-Builder-Konsole zu importieren:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Images aus.
3. Klicken Sie auf Packet importieren.
4. Geben Sie Details zu den folgenden Abschnitten auf der Seite Bild importieren an. Wählen Sie dann Image importieren aus, wenn Sie fertig sind.

Allgemeines

1. Geben Sie einen eindeutigen Name nfür das Basis-Image an.
2. Geben Sie eine Version für das Basis-Image an. Verwenden Sie das folgende Format: *`major.minor.patch`*.
3. Sie können auch eine optionale Beschreibung für das Basis-Image eingeben.

Basis-Image-Betriebssystem

1. Wählen Sie die Image Operating System (OS)-Option aus, die Ihrer VM-Betriebssystemplattform entspricht.
2. Wählen Sie die Betriebssystemversion, die der Version für Ihre VM entspricht, aus der Liste aus.

VM-Importkonfiguration

Wenn Sie Ihre VM aus der Virtualisierungsumgebung exportieren, erstellt dieser Prozess einen Satz von einer oder mehreren Festplattencontainerdateien. Diese dienen als Snapshots der Umgebung, der Einstellungen und der Daten Ihrer VM. Sie können diese Dateien verwenden, um Ihre VM als

Basis-Image für Ihr Image-Rezept zu importieren. Weitere Informationen zum Importieren von VMs in Image Builder finden Sie unter Importieren und Exportieren von VM-Images.

Gehen Sie wie folgt vor, um den Speicherort Ihrer Importquelle anzugeben:

Importquelle

Geben Sie im Abschnitt Datenträgercontainer 1 die Quelle für den ersten VM-Image-Datenträgercontainer oder -Snapshot an, der importiert werden soll.

1. Quelle – Dies kann entweder ein S3-Bucket oder ein EBS-Snapshot sein.
2. S3-Speicherort des Datenträgers auswählen – Geben Sie den Speicherort in Amazon S3 ein, an dem Ihre Datenträgerabbilder gespeichert sind. Um nach dem Speicherort zu suchen, wählen Sie S3 durchsuchen aus.
3. Um einen Datenträgercontainer hinzuzufügen, wählen Sie Datenträgercontainer hinzufügen aus.

IAM-Rolle

Um Ihrer VM-Importkonfiguration eine IAM-Rolle zuzuordnen, wählen Sie die Rolle aus der Dropdown-Liste IAM-Rolle aus oder wählen Sie Neue Rolle erstellen, um eine neue zu erstellen. Wenn Sie eine neue Rolle erstellen, wird die Konsolenseite für IAM-Rollen in einer separaten Registerkarte geöffnet.

Erweiterte Einstellungen – optional

Die folgenden Einstellungen sind optional. Mit diesen Einstellungen können Sie Verschlüsselung, Lizenzierung, Tags und mehr für das vom Import erstellte Basis-Image konfigurieren.

Basis-Image-Architektur

Um die Architektur Ihrer VM-Importquelle anzugeben, wählen Sie einen Wert aus der Liste Architektur aus.

Verschlüsselung

Wenn Ihre VM-Festplattenabbilder verschlüsselt sind, müssen Sie einen Schlüssel angeben, der für den Importvorgang verwendet werden soll. Um einen KMS-Schlüssel für den Import anzugeben, wählen Sie einen Wert aus der Liste Verschlüsselung (KMS-Schlüssel) aus. Die Liste enthält KMS-Schlüssel, auf die Ihr Konto in der aktuellen Region zugreifen kann.

Lizenzverwaltung

Wenn Sie eine VM importieren, erkennt der Importvorgang automatisch das VM-Betriebssystem und wendet die entsprechende Lizenz auf das Basis-Image an. Abhängig von Ihrer Betriebssystemplattform lauten die Lizenztypen wie folgt:

- Lizenz enthalten – Eine entsprechende AWS Lizenz für Ihre Plattform wird auf Ihr Basis-Image angewendet.
- Bring Your Own License (BYOL) – Beibehalten der Lizenz von Ihrer VM, falls zutreffend.

Um Lizenzkonfigurationen, die mit erstellt wurden, AWS License Manager an Ihr Basis-Image anzuhängen, wählen Sie aus der Liste Name der Lizenzkonfiguration aus. Weitere Informationen zu License Manager finden Sie unter Arbeiten mit AWS License Manager

> **Note**
>
> - Lizenzkonfigurationen enthalten Lizenzregeln, die auf den Bedingungen Ihrer Unternehmensvereinbarungen basieren.
> - Linux unterstützt nur BYOL-Lizenzen.

Tags (Basis-Image)

Tags verwenden Schlüssel-Wert-Paare, um Ihrer Image-Builder-Ressource durchsuchbaren Text zuzuweisen. Um Tags für das importierte Basis-Image anzugeben, geben Sie Schlüssel-Wert-Paare mithilfe der Felder Schlüssel und Wert ein.

Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Klicken Sie zum Entfernen eines Tags auf Tag entfernen.

Importieren einer VM (AWS CLI)

Gehen Sie folgendermaßen vor, um eine VM von Datenträgern in ein AMI zu importieren und eine Image Builder-Image-Ressource zu erstellen, auf die Sie sofort verweisen könnenAWS CLI:

1. Initiieren Sie einen VM-Import mit dem Befehl Amazon EC2 VM Import/Export import-image in der AWS CLI. Notieren Sie sich die Aufgaben-ID, die in der Befehlsantwort zurückgegeben wird. Sie benötigen ihn für den nächsten Schritt. Weitere Informationen finden Sie unter Importieren einer VM als Abbild mit VM Import/Export im Benutzerhandbuch für VM Import/Export.

2. Erstellen einer CLI-Eingabe-JSON-Datei

Um den Image-Builder-import-vm-imageBefehl zu optimieren, der in der verwendet wirdAWS CLI, erstellen wir eine JSON-Datei, die alle Importkonfigurationen enthält, die wir an den Befehl übergeben möchten.

> **Note**
>
> Die Namenskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Anforderungsparameter der Image-Builder-API-Aktion angegeben ist. Informationen zum Überprüfen der API-Befehlsanforderungsparameter finden Sie unter dem ImportVmImage Befehl in der EC2 Image Builder-API-Referenz .
> Um die Datenwerte als Befehlszeilenparameter bereitzustellen, beziehen Sie sich auf die Parameternamen, die in der AWS CLI -Befehlsreferenz angegeben sind. auf den Image-Builder-import-vm-imageBefehl als Optionen.

Im Folgenden finden Sie eine Zusammenfassung der Parameter, die wir in diesem Beispiel angeben:

- Name (Zeichenfolge, erforderlich) – Der Name für die Image-Builder-Image-Ressource, die als Ausgabe des Imports erstellt werden soll.
- semanticVersion (Zeichenfolge, erforderlich) – Die semantische Version für das Ausgabebild, die die Version im folgenden Format angibt, mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben: <major>.<minor>.<patch>. Beispiel: `1.0.0` Weitere Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.
- description (Zeichenfolge) – Die Beschreibung des Image-Rezepts.
- -Plattform (Zeichenfolge, erforderlich) – Die Betriebssystemplattform für die importierte VM.
- vmImportTaskID (Zeichenfolge, erforderlich) – Die `ImportTaskId` (AWS CLI) aus dem Amazon EC2 VM-Importprozess. Image Builder überwacht den Importvorgang, um das erstellte AMI abzurufen und eine Image-Builder-Image-Ressource zu erstellen, die sofort in Rezepten verwendet werden kann.
- clientToken (Zeichenfolge, erforderlich) – Eine eindeutige Kennung, bei der zwischen Groß- und Kleinschreibung unterschieden wird, die Sie angeben, um die Idempotenz der

Anforderung sicherzustellen. Weitere Informationen finden Sie unter Sicherstellen von Idempotenz in der Amazon EC2-API-Referenz.

- Tags (Zeichenfolgenzuordnung) – Tags sind Schlüssel-Wert-Paare, die den Importressourcen angefügt sind. Es sind bis zu 50 Schlüssel-Wert-Paare zulässig.

Speichern Sie die Datei als `import-vm-image.json`, um sie im Image-Builder-import-vm-imageBefehl zu verwenden.

```
{
    "name": "example-request",
    "semanticVersion": "1.0.0",
    "description": "vm-import-test",
    "platform": "Linux",
    "vmImportTaskId": "import-ami-01ab234567890cd1e",
    "clientToken": "asz1231231234cs3z",
    "tags": {
     "Usage": "VMIE"
    }
}
```

3. Importieren des Images

Führen Sie den import-vm-image Befehl mit der Datei aus, die Sie als Eingabe erstellt haben:

```
aws imagebuilder import-vm-image --cli-input-json file://import-vm-image.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Verwalten von Sicherheitsergebnissen für Image-Builder-Images

Wenn Sie das Sicherheitsscannen mit Amazon Inspector aktivieren, scannt es kontinuierlich Maschinen-Images und laufende Instances in Ihrem Konto auf Schwachstellen im Betriebssystem und in der Programmiersprache. Wenn diese Option aktiviert ist, erfolgt das Sicherheitsscannen automatisch und Image Builder kann einen Snapshot der Ergebnisse aus Ihrer Test-Instance speichern, wenn Sie ein neues Image erstellen. Amazon Inspector ist ein kostenpflichtiger Service.

Wenn Amazon Inspector Schwachstellen in Ihren Software- oder Netzwerkeinstellungen erkennt, werden die folgenden Aktionen ausgeführt:

- Benachrichtigt Sie, dass ein Ergebnis vorliegt.
- Bewertet den Schweregrad der Erkenntnis. Die Schweregradbewertung kategorisiert Schwachstellen, um Ihnen bei der Priorisierung Ihrer Ergebnisse zu helfen, und enthält die folgenden Werte:
  - Nicht verteilt
  - Informativ
  - Niedrig
  - Medium
  - Hoch
  - Kritisch
- Enthält Informationen über die Erkenntnis und Links zu zusätzlichen Ressourcen für weitere Details.
- Bietet Anleitungen zur Behebung von Problemen, die das Ergebnis generiert haben.

Konfigurieren von Sicherheitsscans für Image-Builder-Images in der AWS Management Console

Wenn Sie Amazon Inspector für Ihr Konto aktiviert haben, scannt Amazon Inspector automatisch die EC2-Instances, die Image Builder startet, um ein neues Image zu erstellen und zu testen. Diese Instances haben während des Build- und Testprozesses eine kurze Lebensdauer und ihre Ergebnisse laufen normalerweise ab, sobald diese Instances heruntergefahren werden. Um Sie bei der Untersuchung und Behebung von Erkenntnissen für Ihr neues Image zu unterstützen, kann Image Builder optional alle Erkenntnisse, die Amazon Inspector während des Erstellungsprozesses auf Ihrer Test-Instance identifiziert hat, als Snapshot speichern.

Schritt 1: Aktivieren von Amazon Inspector-Sicherheitsscans für Ihr Konto

Gehen Sie folgendermaßen vor, um Amazon-Amazon Inspector-Sicherheitsscans für Ihr Konto über die Image-Builder-Konsole zu aktivieren:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Einstellungen für das Scannen von Sicherheit aus. Dadurch wird das Dialogfeld Scannen der Sicherheit geöffnet.

   Das Dialogfeld zeigt den Scanstatus für Ihr Konto an. Wenn Amazon Inspector bereits für Ihr Konto aktiviert ist, zeigt der Status Aktiviert an.
3. Folgen Sie den Schritten 1 und 2 der Anweisungen, um das Scannen von Amazon Inspector zu aktivieren.

   > **Note**
   >
   > Für Amazon Inspector fallen Gebühren an. Weitere Informationen erhalten Sie unter Amazon Inspector: Preise.

Wenn Sie das Scannen für Ihre Pipeline aktiviert haben, erstellt Image Builder einen Snapshot der Ergebnisse für Ihre Build-Instance, wenn Sie ein neues Image erstellen. Auf diese Weise können Sie auf die Ergebnisse zugreifen, nachdem Image Builder die Build-Instance beendet hat.

Schritt 2: Konfigurieren Ihrer Pipeline zum Speichern von Snapshots für Schwachstellenergebnisse

Führen Sie die folgenden Schritte aus, um Snapshots der Schwachstellensuche für Ihre Pipeline zu konfigurieren:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Pipelines aus.
3. Wählen Sie eine der folgenden Methoden, um Pipeline-Details anzugeben:

   Erstellen einer neuen Pipeline

   1. Wählen Sie auf der Seite Image-Pipelines die Option Image-Pipeline erstellen aus. Dadurch wird die Seite Pipeline-Details angeben im Pipeline-Assistenten geöffnet.

Aktualisieren einer vorhandenen Pipeline

1. Wählen Sie auf der Seite Image-Pipelines den Link Pipeline-Name für die Pipeline aus, die Sie aktualisieren möchten. Dadurch wird die Pipeline-Detailseite geöffnet.

   > **Note**
   >
   > Alternativ können Sie das Kontrollkästchen neben dem Namen der Pipeline aktivieren, die Sie aktualisieren möchten, und dann Details anzeigen auswählen.

2. Wählen Sie auf der Seite mit den Pipeline-Details im Menü Aktion die Option Pipeline bearbeiten aus. Dadurch gelangen Sie zur Seite Pipeline bearbeiten.

4. Aktivieren Sie im Abschnitt Allgemein des Pipeline-Assistenten oder auf der Seite Pipeline bearbeiten das Kontrollkästchen Sicherheitsscan aktivieren.

   > **Note**
   >
   > Wenn Sie die Snapshots später deaktivieren möchten, können Sie Ihre Pipeline bearbeiten, um das Kontrollkästchen zu deaktivieren. Dadurch wird das Scannen von Amazon Inspector für Ihr Konto nicht deaktiviert. Informationen zum Deaktivieren des Scannens von Amazon Inspector finden Sie unter Deaktivieren von Amazon Inspector im Amazon Inspector-Benutzerhandbuch.

Verwalten von Sicherheitsergebnissen für Image-Builder-Images in der AWS Management Console

Auf den Listenseiten mit Sicherheitserkenntnissen werden allgemeine Informationen zu den Erkenntnissen für Ihre Ressourcen angezeigt, wobei Ansichten auf der Grundlage mehrerer Filter erstellt werden, die Sie anwenden können. Jede Ansicht enthält oben die folgenden Optionen, um Ihre Ansicht zu ändern:

- Alle Sicherheitserkenntnisse – Dies ist die Standardansicht, wenn Sie im Navigationsbereich der Image-Builder-Konsole die Option Seite Sicherheitserkenntnisse auswählen.
- Nach Schwachstelle – Diese Ansicht zeigt eine allgemeine Liste aller Image-Ressourcen in Ihrem Konto, die Ergebnisse enthalten. Die Erkenntnis-ID ist mit detaillierteren Informationen über die

Erkenntnis verknüpft. Diese Informationen werden in einem Bereich angezeigt, der sich auf der rechten Seite öffnet. Das Panel enthält die folgenden Informationen:

- Eine detaillierte Beschreibung der Erkenntnis.
- Eine Registerkarte mit den Erkenntnisdetails. Diese Registerkarte enthält eine Übersicht über die Erkenntnisse, betroffene Pakete, zusammenfassende Hinweise zur Behebung von Problemen, Schwachstellendetails und zugehörige Schwachstellen. Die Schwachstellen-ID verweist auf detaillierte Schwachstelleninformationen in der National Vulnerability Database.
- Eine Registerkarte Ergebnisaufschlüsselung. Diese Registerkarte enthält einen side-by-side Vergleich der CVSS- und Amazon Inspector-Werte, sodass Sie sehen können, wo Amazon Inspector gegebenenfalls einen Wert geändert hat.

- Nach Image-Pipeline – Diese Ansicht zeigt die Anzahl der Ergebnisse für jede Image-Pipeline in Ihrem Konto. Image Builder zeigt Zählungen für mittleren Schweregrad und höhere Erkenntnisse sowie eine Summe für alle Erkenntnisse an. Alle Daten in der Liste sind wie folgt verknüpft:
  - Die Spalte Image-Pipeline-Name ist mit der Detailseite für die angegebene Image-Pipeline verknüpft.
  - In den Spaltenlinks mit dem Schweregrad wird die Ansicht Alle Sicherheitserkenntnisse geöffnet, gefiltert nach dem Namen der zugehörigen Image-Pipeline und dem Schweregrad.

  Sie können auch Suchkriterien verwenden, um Ihre Ergebnisse zu verfeinern.
- Nach Image – Diese Ansicht zeigt die Anzahl der Ergebnisse für jeden Image-Build in Ihrem Konto. Image Builder zeigt Zählungen für mittleren Schweregrad und höhere Erkenntnisse sowie eine Summe für alle Erkenntnisse an. Alle Daten in der Liste sind wie folgt verknüpft:
  - Die Spalte Image name ist mit der Image-Detailseite für den angegebenen Image-Build verknüpft. Weitere Informationen finden Sie unter Anzeigen von Bilddetails.
  - In den Spaltenlinks mit dem Schweregrad wird die Ansicht Alle Sicherheitserkenntnisse geöffnet, gefiltert nach dem zugehörigen Image-Build-Namen und Schweregrad.

  Sie können auch Suchkriterien verwenden, um Ihre Ergebnisse zu verfeinern.

Image Builder zeigt die folgenden Details im Abschnitt Erkenntnisliste der standardmäßigen Ansicht Alle Sicherheitserkenntnisse an.

Schweregrad

Der Schweregrad der CVE-Erkenntnis. Werte sind wie folgt:

- Nicht verteilt

- Informativ
- Niedrig
- Medium
- Hoch
- Kritisch

Die ID des Ergebnisses

Die eindeutige Kennung für die CVE-Erkenntnis, die Amazon Inspector beim Scannen der Build-Instance für Ihr Image erkannt hat. Die ID ist mit der Seite Sicherheitserkenntnisse > Nach Schwachstelle verknüpft.

Image-ARN

Der Amazon-Ressourcenname (ARN) für das Bild mit der Erkenntnis, die in der Spalte Erkenntnis-ID angegeben ist.

Pipeline

Die Pipeline, die das in der Spalte Image ARN angegebene Image erstellt hat.

Beschreibung

Eine kurze Beschreibung der Erkenntnis.

Inspector-Score

Der Wert, den Amazon Inspector für die CVE-Erkenntnis zugewiesen hat.

Behebung

Links zu Details über die empfohlene Vorgehensweise zur Behebung der Erkenntnis.

Veröffentlichungsdatum

Das Datum und die Uhrzeit, zu der diese Schwachstelle zum ersten Mal zur Datenbank des Anbieters hinzugefügt wurde.

Bereinigen von -Ressourcen

Um unerwartete Gebühren zu vermeiden, stellen Sie sicher, dass Sie Ressourcen und Pipelines bereinigen, die Sie aus den Beispielen in diesem Handbuch erstellt haben. Weitere Informationen

zum Löschen von Ressourcen in Image Builder finden Sie unter Löschen von EC2 Image Builder-Ressourcen.

Verwalten der Infrastrukturkonfiguration von EC2 Image Builder

Sie können Infrastrukturkonfigurationen verwenden, um die Amazon EC2-Infrastruktur anzugeben, die Image Builder zum Erstellen und Testen Ihres EC2-Image-Builder-Images verwendet. Zu den Infrastruktureinstellungen gehören:

- Instance-Typen für Ihre Build- und Testinfrastruktur. Wir empfehlen, mehr als einen Instance-Typ anzugeben, da Image Builder dadurch eine Instance aus einem Pool mit ausreichender Kapazität starten kann. Dies kann Ihre vorübergehenden Build-Fehler reduzieren.
- Ein Instance-Profil, das Ihren Build- und Test-Instances die Berechtigungen bereitstellt, die für die Durchführung von Anpassungsaktivitäten erforderlich sind. Wenn Sie beispielsweise eine Komponente haben, die Ressourcen von Amazon S3 abruft, benötigt das Instance-Profil Berechtigungen für den Zugriff auf diese Dateien. Das Instance-Profil benötigt auch einen minimalen Satz von Berechtigungen, damit EC2 Image Builder erfolgreich mit der Instance kommunizieren kann. Weitere Informationen finden Sie unter Voraussetzungen.
- Die VPC, das Subnetz und die Sicherheitsgruppen für die Build- und Test-Instances Ihrer Pipeline.
- Der Amazon S3-Speicherort, an dem Image Builder Anwendungsprotokolle aus Ihrem Build und Ihren Tests speichert. Wenn Sie die Protokollierung konfigurieren, muss das in Ihrer Infrastrukturkonfiguration angegebene Instance-Profil über `s3:PutObject` Berechtigungen für den Ziel-Bucket () verfügen`arn:aws:s3:::BucketName/*`.
- Ein Amazon EC2-Schlüsselpaar, mit dem Sie sich bei Ihrer Instance anmelden können, um Fehler zu beheben, wenn Ihr Build fehlschlägt und Sie `terminateInstanceOnFailure` auf festlegen`false`.
- Ein SNS-Thema, an das Image Builder Ereignisbenachrichtigungen sendet. Weitere Informationen zur Integration von Image Builder in Amazon SNS finden Sie unter Amazon SNS-Integration in Image Builder.

> **Note**
>
> Wenn Ihr SNS-Thema verschlüsselt ist, muss sich der Schlüssel, der dieses Thema verschlüsselt, in dem Konto befinden, in dem der Image-Builder-Service ausgeführt wird. Image Builder kann keine Benachrichtigungen an SNS-Themen senden, die mit Schlüsseln von anderen Konten verschlüsselt sind.

Sie können Infrastrukturkonfigurationen mithilfe der Image-Builder-Konsole, über die Image-Builder-API oder mit -imagebuilderBefehlen in der erstellen und verwaltenAWS CLI.

Inhalt

> **Tip**
>
> Wenn Sie mehrere Ressourcen desselben Typs haben, hilft Ihnen das Markieren dabei, eine bestimmte Ressource anhand der ihr zugewiesenen Tags zu identifizieren. Weitere Informationen zum Markieren Ihrer Ressourcen mit Image-Builder-Befehlen in der AWS CLIfinden Sie im Markieren von Ressourcen Abschnitt dieses Handbuchs.

Auflisten und Anzeigen von Infrastrukturkonfigurationsdetails

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder-Infrastrukturkonfigurationen anzeigen können.

Details zur Infrastrukturkonfiguration

Auflisten von Infrastrukturkonfigurationen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie alle Ihre Infrastrukturkonfigurationen mit dem list-infrastructure-configurations Befehl in der auflistenAWS CLI.

```
aws imagebuilder list-infrastructure-configurations
```

Details zur Infrastrukturkonfiguration abrufen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den get-infrastructure-configuration Befehl in der verwendenAWS CLI, um die Details einer Infrastrukturkonfiguration abzurufen, indem Sie ihren Amazon-Ressourcennamen (ARN) angeben.

```
aws imagebuilder get-infrastructure-configuration --infrastructure-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-
infrastructure-configuration
```

Erstellen einer Infrastrukturkonfiguration

In diesem Abschnitt wird beschrieben, wie Sie die Image-Builder-Konsole oder imagebuilder Befehle in der verwenden könnenAWS CLI, um eine Infrastrukturkonfiguration zu erstellen.

Console

Gehen Sie folgendermaßen vor, um eine Infrastrukturkonfigurationsressource über die Image-Builder-Konsole zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
3. Wählen Sie Infrastrukturkonfiguration erstellen aus.
4. Geben Sie im Abschnitt Allgemein die folgenden erforderlichen Informationen ein:
   - Geben Sie den Name nIhrer Infrastrukturkonfigurationsressource ein.
   - Wählen Sie eine IAM-Rolle aus, die Sie dem Instance-Profil für Komponentenberechtigungen auf Ihren Build- und Test-Instances zuordnen möchten. Image Builder verwendet diese Berechtigungen, um Ihre Komponenten herunterzuladen und auszuführen, Protokolle in hochzuladen und zusätzliche Aktionen auszuführen CloudWatch, die die Komponenten in Ihrem Rezept angeben.
5. Im AWS Infrastrukturbereich können Sie alle verbleibenden verfügbaren Infrastruktureinstellungen konfigurieren. Geben Sie die folgenden erforderlichen Informationen ein:

- Instance-Typ – Sie können einen oder mehrere Instance-Typen angeben, die für diesen Build verwendet werden sollen. Der Service wählt je nach Verfügbarkeit einen dieser Instance-Typen aus.

Wenn Sie keine Werte für die folgenden Einstellungen angeben, verwenden diese gegebenenfalls servicespezifische Standardwerte.

- VPC, Subnetz und Sicherheitsgruppen – Image Builder verwendet Ihre Standard-VPC und Ihr Subnetz. Weitere Informationen zum Konfigurieren von VPC-Schnittstellenendpunkten finden Sie unter EC2 Image Builder und Schnittstellen-VPC-Endpunkte (AWS PrivateLink).
- Im Abschnitt Fehlerbehebungseinstellungen können Sie die folgenden Werte konfigurieren:
  - Standardmäßig ist das Kontrollkästchen Instance bei Fehler beenden aktiviert. Wenn ein Build jedoch fehlschlägt, können Sie sich zur Fehlerbehebung bei der EC2-Instance anmelden. Wenn Ihre Instance nach einem Build-Fehler weiter ausgeführt werden soll, deaktivieren Sie das Kontrollkästchen.
  - Schlüsselpaar – Wenn Ihre EC2-Instance nach einem Build-Fehler weiter ausgeführt wird, können Sie ein Schlüsselpaar erstellen oder ein vorhandenes Schlüsselpaar verwenden, um sich bei der Instance anzumelden und Fehler zu beheben.
  - Protokolle – Sie können einen S3-Bucket angeben, in den Image Builder Anwendungsprotokolle schreiben kann, um die Fehlerbehebung bei Ihren Builds und Tests zu unterstützen. Wenn Sie keinen S3-Bucket angeben, schreibt Image Builder die Anwendungsprotokolle in die Instance.
- Im Abschnitt Einstellungen für Instance-Metadaten können Sie die folgenden Werte so konfigurieren, dass sie auf die EC2-Instances angewendet werden, die Image Builder zum Erstellen und Testen Ihres Images verwendet:
  - Wählen Sie die Metadatenversion aus, um festzustellen, ob EC2 einen signierten Token-Header für Abrufanforderungen für Instance-Metadaten benötigt.
    - V1 und V2 (Token optional) – Standardwert, wenn Sie nichts auswählen.
    - V2 (Token erforderlich)

> **Note**
>
> Wir empfehlen, alle EC2-Instances, die Image Builder aus einem Pipeline-Build startet, für die Verwendung von IMDSv2 zu konfigurieren, damit

Abrufanforderungen für Instance-Metadaten einen signierten Token-Header erfordern.

- Metadaten-Token-Antwort-Hop-Limit – Die Anzahl der Netzwerk-Hops, die das Metadaten-Token zurücklegen kann. Minimale Hops: 1, maximale Hops: 64, mit einem Standardwert von einem Hop.

AWS CLI

Das folgende Beispiel zeigt, wie Sie die Infrastruktur für Ihr Image mit dem Image-Buildercreate-infrastructure-configuration-Befehl in konfigurierenAWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Dieses Beispiel für eine Infrastrukturkonfiguration gibt zwei Instance-Typen an: `m5.large` und `m5.xlarge`. Wir empfehlen, mehr als einen Instance-Typ anzugeben, da Image Builder dadurch eine Instance aus einem Pool mit ausreichender Kapazität starten kann. Dies kann Ihre vorübergehenden Build-Fehler reduzieren.

   gibt das Instance-Profil `instanceProfileName` an, das der Instance die Berechtigungen erteilt, die das Profil zum Durchführen von Anpassungsaktivitäten benötigt. Wenn Sie beispielsweise eine Komponente haben, die Ressourcen von Amazon S3 abruft, benötigt das Instance-Profil Berechtigungen für den Zugriff auf diese Dateien. Das Instance-Profil benötigt auch einen minimalen Satz von Berechtigungen, damit EC2 Image Builder erfolgreich mit der Instance kommunizieren kann. Weitere Informationen finden Sie unter Voraussetzungen.

   Verwenden Sie ein Dateibearbeitungstool, um eine JSON-Datei mit Schlüsseln zu erstellen, die im folgenden Beispiel gezeigt werden, sowie Werte, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-infrastructure-configuration.json`:

```
{
    "name": "MyExampleInfrastructure",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
        "m5.large", "m5.xlarge"
    ],
    "instanceProfileName": "myIAMInstanceProfileName",
    "securityGroupIds": [
```

```
        "sg-12345678"
    ],
    "subnetId": "sub-12345678",
    "logging": {
        "s3Logs": {
            "s3BucketName": "my-logging-bucket",
            "s3KeyPrefix": "my-path"
        }
    },
    "keyPair": "myKeyPairName",
    "terminateInstanceOnFailure": false,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

2. Verwenden Sie die Datei, die Sie als Eingabe erstellt haben, wenn Sie den folgenden Befehl ausführen.

```
aws imagebuilder create-infrastructure-configuration --cli-input-json
 file://create-infrastructure-configuration.json
```

Aktualisieren einer Infrastrukturkonfiguration

In diesem Abschnitt wird beschrieben, wie Sie die Image-Builder-Konsole oder imagebuilder Befehle in der verwenden könnenAWS CLI, um eine Infrastrukturkonfigurationsressource zu aktualisieren.

Console

Sie können die folgenden Infrastrukturkonfigurationsdetails über die Image-Builder-Konsole bearbeiten:

- Die Beschreibung für Ihre Infrastrukturkonfiguration.
- Die IAM-Rolle, die dem Instance-Profil zugeordnet werden soll.
- AWS Infrastruktur , einschließlich des Instance-Typs und eines SNS-Themas für Benachrichtigungen.
- VPC, Subnetz und Sicherheitsgruppen .
- Fehlerbehebungseinstellungen, einschließlich Beenden der Instance bei Fehlern, das Schlüsselpaar für die Verbindung und ein optionaler S3-Bucket-Speicherort für Instance-Protokolle.

Gehen Sie folgendermaßen vor, um eine Infrastrukturkonfigurationsressource über die Image-Builder-Konsole zu aktualisieren:

Wählen Sie eine vorhandene Image-Builder-Infrastrukturkonfiguration

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der Infrastrukturkonfigurationsressourcen unter Ihrem Konto anzuzeigen, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
3. Um Details anzuzeigen oder eine Infrastrukturkonfiguration zu bearbeiten, wählen Sie den Link Konfigurationsname. Dadurch wird die Detailansicht für die Infrastrukturkonfiguration geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Konfigurationsnamen aktivieren und dann Details anzeigen auswählen.

4. Wählen Sie oben rechts im Bereich Infrastrukturdetails die Option Bearbeiten aus.
5. Wenn Sie bereit sind, Aktualisierungen zu speichern, die Sie an Ihrer Infrastrukturkonfiguration vorgenommen haben, wählen Sie Änderungen speichern aus.

AWS CLI

Das folgende Beispiel zeigt, wie Sie die Infrastrukturkonfiguration für Ihr Image mit dem Image-Builderupdate-infrastructure-configuration-Befehl in aktualisierenAWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   In diesem Beispiel für die Infrastrukturkonfiguration werden dieselben Einstellungen wie im Erstellungsbeispiel verwendet, mit der Ausnahme, dass wir die `terminateInstanceOnFailure` Einstellung auf aktualisiert haben`false`. Nachdem wir den update-infrastructure-configuration Befehl ausgeführt haben, beenden Pipelines, die diese Infrastrukturkonfiguration verwenden, die Build- und Test-Instances, wenn der Build fehlschlägt.

   Verwenden Sie ein Dateibearbeitungstool, um eine JSON-Datei mit Schlüsseln zu erstellen, die im folgenden Beispiel gezeigt werden, sowie Werte, die für Ihre Umgebung gültig

sind. In diesem Beispiel wird eine Datei mit dem Namen`update-infrastructure-configuration.json`:

```
{
"infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
"description": "An example that will terminate instances of failed builds",
"instanceTypes": [
    "m5.large", "m5.2xlarge"
],
"instanceProfileName": "myIAMInstanceProfileName",
"securityGroupIds": [
    "sg-12345678"
],
"subnetId": "sub-12345678",
"logging": {
    "s3Logs": {
        "s3BucketName": "my-logging-bucket",
        "s3KeyPrefix": "my-path"
    }
},
"terminateInstanceOnFailure": true,
"snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

2. Verwenden Sie die Datei, die Sie als Eingabe erstellt haben, wenn Sie den folgenden Befehl ausführen.

```
aws imagebuilder update-infrastructure-configuration --cli-input-json
 file://update-infrastructure-configuration.json
```

EC2 Image Builder und Schnittstellen-VPC-Endpunkte (AWS PrivateLink)

Sie können eine private Verbindung zwischen Ihrer VPC und EC2 Image Builder herstellen, indem Sie einen Schnittstellen-VPC-Endpunkt erstellen. Schnittstellenendpunkte werden von unterstütztAWS PrivateLink, einer Technologie, mit der Sie ohne Internet-Gateway, NAT-Gerät, VPN-Verbindung oder -AWS Direct ConnectVerbindung privat auf Image-Builder-APIs zugreifen können. Instances in Ihrer VPC benötigen für die Kommunikation mit Image-Builder-APIs keine

öffentlichen IP-Adressen. Der Datenverkehr zwischen Ihrer VPC und Image Builder verlässt das Amazon-Netzwerk nicht.

Jeder Schnittstellenendpunkt wird durch eine oder mehrere Elastic-Network-Schnittstellen in Ihren Subnetzen dargestellt. Wenn Sie ein neues Image erstellen, können Sie die VPC-Subnetz-ID in Ihrer Infrastrukturkonfiguration angeben.

> **Note**
>
> Jeder Service, auf den Sie von einer VPC aus zugreifen, hat seinen eigenen Schnittstellenendpunkt mit einer eigenen Endpunktrichtlinie. Image Builder lädt die AWSTOE component Manager-Anwendung herunter und greift auf verwaltete Ressourcen aus S3-Buckets zu, um benutzerdefinierte Images zu erstellen. Um Zugriff auf diese Buckets zu gewähren, müssen Sie die S3-Endpunktrichtlinie aktualisieren, um dies zu erlauben. Weitere Informationen finden Sie unter Benutzerdefinierte Richtlinien für den S3-Bucket-Zugriff.

Weitere Informationen zu VPC-Endpunkten finden Sie unter Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon VPC Benutzerhandbuch.

Überlegungen zu Image-Builder-VPC-Endpunkten

Bevor Sie einen Schnittstellen-VPC-Endpunkt für Image Builder einrichten, lesen Sie die Eigenschaften und Einschränkungen von Schnittstellenendpunkten im Amazon-VPC-Benutzerhandbuch.

Image Builder unterstützt Aufrufe aller API-Aktionen von Ihrer VPC aus.

Erstellen eines Schnittstellen-VPC-Endpunkts für Image Builder

Um einen VPC-Endpunkt für den Image-Builder-Service zu erstellen, können Sie entweder die Amazon-VPC-Konsole oder die AWS Command Line Interface (AWS CLI) verwenden. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Benutzerhandbuch für Amazon VPC.

Erstellen Sie einen VPC-Endpunkt für Image Builder mit dem folgenden Servicenamen:

- com.amazonaws.*region*.imagebuilder

Wenn Sie ein privates DNS für den Endpunkt aktivieren, können Sie API-Anforderungen an Image Builder unter Verwendung seines Standard-DNS-Namens für die Region senden, z. B.: `imagebuilder.us-east-1.amazonaws.com`. Informationen zum Suchen des Endpunkts, der für Ihre Zielregion gilt, finden Sie unter EC2 Image Builder-Endpunkte und -Kontingente im Allgemeine Amazon Web Services-Referenz.

Weitere Informationen finden Sie unter Zugriff auf einen Service über einen Schnittstellenendpunkt im Benutzerhandbuch für Amazon VPC.

Erstellen einer VPC-Endpunktrichtlinie für Image Builder

Sie können Ihrem VPC-Endpunkt eine Endpunktrichtlinie anfügen, die den Zugriff auf Image Builder steuert. Die Richtlinie gibt die folgenden Informationen an:

- Prinzipal, der die Aktionen ausführen kann.
- Aktionen, die ausgeführt werden können
- Die Ressourcen, für die Aktionen ausgeführt werden können.

Wenn Sie von Amazon verwaltete Komponenten in Ihrem Rezept verwenden, muss der VPC-Endpunkt für Image Builder den Zugriff auf die folgende serviceeigene Komponentenbibliothek zulassen:

`arn:aws:imagebuilder:region:aws:component/*`

> **Important**
>
> Wenn eine nicht standardmäßige Richtlinie auf einen Schnittstellen-VPC-Endpunkt für EC2 Image Builder angewendet wird, werden bestimmte API-Anforderungen, z. B. solche, die von fehlschlagen`RequestLimitExceeded`, möglicherweise nicht in AWS CloudTrail oder Amazon protokolliert CloudWatch.

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

Benutzerdefinierte Richtlinien für den S3-Bucket-Zugriff

Image Builder verwendet einen öffentlich verfügbaren S3-Bucket, um verwaltete Ressourcen wie Komponenten zu speichern und darauf zuzugreifen. Außerdem wird die Anwendung zur AWSTOE

Komponentenverwaltung aus einem separaten S3-Bucket heruntergeladen. Wenn Sie einen VPC-Endpunkt für Amazon S3 in Ihrer Umgebung verwenden, müssen Sie sicherstellen, dass Ihre S3-VPC-Endpunktrichtlinie Image Builder den Zugriff auf die folgenden S3-Buckets ermöglicht. Die Bucket-Namen sind pro AWS Region (*Region*) und Anwendungsumgebung (*Umgebung*) eindeutig. Image Builder und AWSTOE unterstützen die folgenden Anwendungsumgebungen: `prod`, `preprod`, und `beta`.

- Der AWSTOE Komponentenmanager-Bucket:

  ```
  s3://ec2imagebuilder-toe-region-environment
  ```

  Beispiel: s3://ec2imagebuilder-toe-us-west-2-prod/*

- Der verwaltete Ressourcen-Bucket von Image Builder:

  ```
  s3://ec2imagebuilder-managed-resources-region-environment/components
  ```

  Beispiel: s3://ec2imagebuilder-managed-resources-us-west-2-prod/ componentss/*

Beispiele für VPC-Endpunktrichtlinien

Dieser Abschnitt enthält Beispiele für benutzerdefinierte VPC-Endpunktrichtlinien.

Allgemeine VPC-Endpunktrichtlinie für Image-Builder-Aktionen

Die folgende Beispiel-Endpunktrichtlinie für Image Builder verweigert die Berechtigung zum Löschen von Image-Builder-Images und -Komponenten. Die Beispielrichtlinie gewährt auch die Berechtigung zum Ausführen aller anderen EC2 Image Builder-Aktionen.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Action": "imagebuilder:*",
        "Effect": "Allow",
        "Resource": "*"
    },
    {
        "Action": [
            "imagebuilder: DeleteImage"
```

```
        ],
        "Effect": "Deny",
        "Resource": "*",
    },
    {
        "Action": [
            "imagebuilder: DeleteComponent"
        ],
        "Effect": "Deny",
        "Resource": "*",
    }]
}
```

Beschränken des Zugriffs nach Organisation, Zulassen des Zugriffs auf verwaltete Komponenten

Die folgende Beispiel-Endpunktrichtlinie zeigt, wie Sie den Zugriff auf Identitäten und Ressourcen einschränken, die zu Ihrer Organisation gehören, und Zugriff auf die von Amazon verwalteten AWSTOE Komponenten gewähren. Ersetzen Sie *Region* , *principal-org-id*und *resource-org-id* durch die Werte Ihrer Organisation.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowRequestsByOrgsIdentitiesToOrgsResources",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "principal-org-id",
          "aws:ResourceOrgID": "resource-org-id"
        }
      }
    },
    {
      "Sid": "AllowAccessToEC2ImageBuilderComponents",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
```

```
      },
      "Action": [
        "imagebuilder:GetComponent"
      ],
      "Resource": [
        "arn:aws:imagebuilder:region:aws:component/*"
      ]
    }
  ]
}
```

VPC-Endpunktrichtlinie für den Zugriff auf Amazon S3-Buckets

Das folgende Beispiel für eine S3-Endpunktrichtlinie zeigt, wie Sie Zugriff auf die S3-Buckets gewähren, die Image Builder zum Erstellen benutzerdefinierter Images verwendet. Ersetzen Sie *Region* und *Umgebung* durch die Werte Ihrer Organisation. Fügen Sie der Richtlinie alle anderen erforderlichen Berechtigungen basierend auf Ihren Anwendungsanforderungen hinzu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowImageBuilderAccessToAppAndComponentBuckets",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::ec2imagebuilder-toe-region-environment/*",
        "arn:aws:s3:::ec2imagebuilder-managed-resources-region-environment/components/
*"
      ]
    }
  ]
}
```

Verwalten von EC2 Image Builder-Verteilungseinstellungen

Nachdem Sie Verteilungseinstellungen mit Image Builder erstellt haben, können Sie sie mithilfe der Image-Builder-Konsole, der Image-Builder-API oder der Befehle imagebuilder in der verwaltenAWS CLI. Mit Verteilungseinstellungen können Sie die folgenden Aktionen ausführen:

AMI-Verteilung

- Geben Sie den Namen und die Beschreibung Ihres Ausgabe-AMI an.
- Autorisieren Sie andere AWS-Konten, Organisationen und OUs, das AMI über das Konto des Besitzers zu starten. Dem Besitzerkonto werden die mit dem AMI verbundenen Gebühren in Rechnung gestellt.

  > **Note**
  >
  > Um ein AMI öffentlich zu machen, setzen Sie die Startberechtigung für autorisierte Konten auf `all`. Sehen Sie sich die Beispiele für die Veröffentlichung eines AMI an unter EC2 ModifyImageAttribute.

- Erstellen Sie eine Kopie des Ausgabe-AMI für jedes der angegebenen Zielkonten, Organisationen und OUs in der Zielregion. Die Zielkonten, Organisationen und OUs besitzen ihre AMI-Kopien und werden für alle damit verbundenen Gebühren in Rechnung gestellt. Weitere Informationen zur Verteilung Ihres AMI an AWS Organizations und OUs finden Sie unter Freigeben eines AMI für Organisationen oder OUs.
- Kopieren Sie das AMI in das Konto des Besitzers in anderen AWS-Regionen.
- Exportieren Sie VM-Image-Datenträger nach Amazon Simple Storage Service (Amazon S3). Weitere Informationen finden Sie unter Erstellen von Verteilungseinstellungen für VM-Ausgabedatenträger (AWS CLI).

Container-Image-Verteilung

- Geben Sie das ECR-Repository an, in dem Image Builder das Ausgabe-Image in der Verteilungsregion speichert.

Sie können Ihre Verteilungseinstellungen auf folgende Weise verwenden, um Images einmal oder bei jedem Pipeline-Build an Zielregionen, Konten AWS Organizations und Organisationseinheiten (OUs) zu übermitteln:

- Um automatisch aktualisierte Images an bestimmte Regionen, Konten, Organisationen und OUs zu übermitteln, verwenden Sie Verteilungseinstellungen mit einer Image-Builder-Pipeline, die nach einem Zeitplan ausgeführt wird.
- Um ein neues Image zu erstellen und es an die angegebenen Regionen, Konten, Organisationen und OUs zu übermitteln, verwenden Sie Verteilungseinstellungen mit einer Image-Builder-Pipeline, die Sie einmal über die Image-Builder-Konsole ausführen, indem Sie Pipeline ausführen im Menü Aktionen verwenden.
- Um ein neues Image zu erstellen und es an die angegebenen Regionen, Konten, Organisationen und OUs zu übermitteln, verwenden Sie Verteilungseinstellungen mit der folgenden API-Aktion oder dem Image-Builder-Befehl in der AWS CLI:
  - Die CreateImage Aktion in der Image Builder-API.
  - Der create-image Befehl in der AWS CLI.
- So exportieren Sie Image-Datenträger der virtuellen Maschine (VM) im Rahmen Ihres regulären Image-Build-Prozesses in S3-Buckets in Zielregionen.

> **Tip**
>
> Wenn Sie mehrere Ressourcen desselben Typs haben, hilft Ihnen das Markieren dabei, eine bestimmte Ressource anhand der ihr zugewiesenen Tags zu identifizieren. Weitere Informationen zum Markieren Ihrer Ressourcen mit Image-Builder-Befehlen in der AWS CLIfinden Sie im Markieren von Ressourcen Abschnitt dieses Handbuchs.

In diesem Thema wird beschrieben, wie Sie Verteilungseinstellungen auflisten, anzeigen und erstellen.

Inhalt

Details zu Verteilungseinstellungen auflisten und anzeigen

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder-Verteilungseinstellungen anzeigen können.

Details zu Verteilungseinstellungen

- Auflisten von Verteilungskonfigurationen (Konsole)
- Anzeigen von Verteilungskonfigurationsdetails (Konsole)
- Auflisten von Verteilungen (AWS CLI)
- Abrufen von Verteilungskonfigurationsdetails (AWS CLI)

Auflisten von Verteilungskonfigurationen (Konsole)

Gehen Sie folgendermaßen vor, um eine Liste der Verteilungskonfigurationen anzuzeigen, die unter Ihrem Konto in der Image-Builder-Konsole erstellt wurden:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Verteilungseinstellungen aus. Dies zeigt eine Liste der Verteilungskonfigurationen, die unter Ihrem Konto erstellt werden.
3. Um Details anzuzeigen oder eine neue Verteilungskonfiguration zu erstellen, wählen Sie den Link Konfigurationsname. Dadurch wird die Detailansicht für die Verteilungseinstellungen geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Konfigurationsnamen aktivieren und dann Details anzeigen auswählen.

Anzeigen von Verteilungskonfigurationsdetails (Konsole)

Um Details für eine bestimmte Verteilungskonfiguration mithilfe der Image-Builder-Konsole anzuzeigen, wählen Sie die zu überprüfende Konfiguration aus. Gehen Sie dazu wie unter beschrieben vorAuflisten von Verteilungskonfigurationen (Konsole).

Auf der Seite mit den Verteilungsdetails können Sie:

- Löschen Sie die Verteilungskonfiguration. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unter Löschen von EC2 Image Builder-Ressourcen.
- Bearbeiten Sie die Verteilungsdetails.

Auflisten von Verteilungen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den list-distribution-configurations Befehl in verwenden, AWS CLI um alle Ihre Verteilungen aufzulisten.

```
aws imagebuilder list-distribution-configurations
```

Abrufen von Verteilungskonfigurationsdetails (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den get-distribution-configuration Befehl in der verwenden, AWS CLI um die Details einer Verteilungskonfiguration abzurufen, indem Sie ihren Amazon-Ressourcennamen (ARN) angeben.

```
aws imagebuilder get-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

Erstellen und Aktualisieren von AMI-Verteilungskonfigurationen

In diesem Abschnitt wird das Erstellen und Aktualisieren von Verteilungskonfigurationen für ein Image Builder-AMI behandelt.

Inhalt

Erstellen einer AMI-Verteilungskonfiguration (Konsole)

Zu den Verteilungskonfigurationen gehören der Name des Ausgabe-AMI, spezifische Regionseinstellungen für die Verschlüsselung, Startberechtigungen und AWS-Konten, Organisationen und Organisationseinheiten (OUs), die das Ausgabe-AMI starten können, und Lizenzkonfigurationen.

So erstellen Sie eine neue AMI-Verteilungskonfiguration:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Verteilungseinstellungen aus. Dies zeigt eine Liste der Verteilungskonfigurationen, die unter Ihrem Konto erstellt werden.
3. Wählen Sie oben im Bereich Verteilungseinstellungen die Option Verteilungseinstellungen erstellen aus.
4. Wählen Sie im Abschnitt Image type den Amazon Machine Image (AMI)-Ausgabetyp aus.
5. Geben Sie im Abschnitt Allgemein einen Name nfür Ihre Verteilungskonfiguration und eine optionale Beschreibung ein.
6. Geben Sie im Abschnitt Regionseinstellungen die folgenden Details für jede Region ein, in der Sie Ihr AMI verteilen:

    a. Das AMI wird standardmäßig an die aktuelle Region (Region 1) verteilt. Region 1 ist die Quelle für die Verteilung. Einige Einstellungen für Region 1 sind nicht zur Bearbeitung geöffnet. Für alle Regionen, die Sie hinzufügen, können Sie eine Region aus der Dropdown-Liste Region auswählen.

    Der Kms-Schlüssel identifiziert die AWS KMS key, die zum Verschlüsseln der EBS-Volumes für Ihr Image in der Zielregion verwendet wird. Beachten Sie, dass dies nicht für das ursprüngliche AMI gilt, das der Build unter Ihrem Konto in der Quellregion (Region 1) erstellt. Die Verschlüsselung, die während der Verteilungsphase des Builds ausgeführt wird, gilt nur für Images, die an andere Konten oder Regionen verteilt werden.

    Um die EBS-Volumes für das AMI zu verschlüsseln, das in der Quellregion für Ihr Konto erstellt wurde, müssen Sie den KMS-Schlüssel in der Blockgerät-Zuweisung des Image-Rezepts (Speicher (Volumes) in der Konsole) festlegen.

    Image Builder kopiert das AMI in die Zielkonten, die Sie für die Region angeben.

> **Voraussetzung**
>
> Um ein Image kontenübergreifend zu kopieren, müssen Sie die `EC2ImageBuilderDistributionCrossAccountRole` Rolle in allen Zielkonten in den Zielregionen erstellen und die Ec2ImageBuilderCrossAccountDistributionAccess-Richtlinie verwaltete Richtlinie an die Rolle anfügen.

Der Name des Ausgabe-AMI ist optional. Wenn Sie einen Namen angeben, enthält der Name des endgültigen Ausgabe-AMI einen angehängten Zeitstempel, wann das AMI erstellt wird. Wenn Sie keinen Namen angeben, hängt Image Builder den Build-Zeitstempel an den Rezeptnamen an. Dadurch werden eindeutige AMI-Namen für jeden Build sichergestellt.

i. Mit der AMI-Freigabe können Sie bestimmten AWSPrinzipalen Zugriff gewähren, um Instances von Ihrem AMI aus zu starten. Wenn Sie den Abschnitt AMI-Freigabe erweitern, können Sie die folgenden Details eingeben:

   - Startberechtigungen – Wählen Sie Privat aus, wenn Sie Ihr AMI privat halten und bestimmten AWSPrinzipalen Zugriff gewähren möchten, um eine Instance von Ihrem privaten AMI aus zu starten. Wählen Sie Öffentlich aus, wenn Sie Ihr AMI öffentlich machen möchten. Jeder AWSPrinzipal kann eine Instance von Ihrem öffentlichen AMI aus starten.
   - Prinzipale – Sie können den folgenden AWSPrinzipalen Zugriff gewähren, um Instances zu starten:
     - AWS Konto – Gewähren Sie Zugriff auf ein bestimmtes AWS Konto
     - Organisationseinheit (OU) – Gewähren Sie Zugriff auf eine OU und alle untergeordneten Entitäten. Zu den untergeordneten Entitäten gehören OUs und AWS Konten.
     - Organisation – AWS OrganizationsGewähren Sie Zugriff auf Ihre und alle untergeordneten Entitäten. Zu den untergeordneten Entitäten gehören OUs und AWS Konten.

       Wählen Sie zunächst den Prinzipaltyp aus. Geben Sie dann die ID für den AWSPrinzipal ein, dem Sie Zugriff gewähren möchten, in das Feld rechts neben der Dropdown-Liste. Sie können mehrere IDs verschiedener Typen eingeben.

ii. Sie können den Abschnitt Lizenzkonfiguration erweitern, um Lizenzkonfigurationen, die mit erstellt wurdenAWS License Manager, an Ihre Image-Builder-Images anzuhängen. Lizenzkonfigurationen enthalten Lizenzregeln, die auf den Bedingungen Ihrer Unternehmensvereinbarungen basieren. Image Builder enthält automatisch Lizenzkonfigurationen, die Ihrem Basis-AMI zugeordnet waren.

iii. Sie können den Abschnitt Konfiguration der Startvorlage erweitern, um eine EC2-Startvorlage anzugeben, die zum Starten von Instances über das von Ihnen erstellte AMI verwendet werden soll.

Wenn Sie eine EC2-Startvorlage verwenden, können Sie Image Builder anweisen, eine neue Version Ihrer Startvorlage zu erstellen, die nach Abschluss des Builds die neueste AMI-ID enthält. Um die Startvorlage zu aktualisieren, konfigurieren Sie die Einstellungen wie folgt:

- Name der Startvorlage – Wählen Sie den Namen der Startvorlage aus, die Image Builder aktualisieren soll.
- Festlegen der Standardversion – Aktivieren Sie dieses Kontrollkästchen, um die Standardversion der Startvorlage auf die neue Version zu aktualisieren.

Um eine weitere Startvorlagenkonfiguration hinzuzufügen, wählen Sie Startvorlagenkonfiguration hinzufügen aus. Sie können bis zu fünf Startvorlagenkonfigurationen pro Region haben.

b. Um Verteilungseinstellungen für eine andere Region hinzuzufügen, wählen Sie Region hinzufügen aus.

7. Wählen Sie anschließend Einstellungen erstellen aus.

Erstellen von Verteilungseinstellungen für Ausgabe-AMIs (AWS CLI)

Mit einer Verteilungskonfiguration können Sie den Namen und die Beschreibung Ihres Ausgabe-AMI angeben, andere AWS-Konten zum Starten des AMI autorisieren, das AMI in andere Konten kopieren und das AMI in andere AWS Regionen replizieren. Außerdem können Sie das AMI nach Amazon Simple Storage Service (Amazon S3) exportieren oder EC2 Fast Launch für die Ausgabe von Windows-AMIs konfigurieren. Um ein AMI öffentlich zu machen, setzen Sie die Startberechtigung für autorisierte Konten auf `all`. Beispiele für die Veröffentlichung eines AMI finden Sie unter EC2 ModifyImageAttribute.

Das folgende Beispiel zeigt, wie Sie mit dem create-distribution-configuration Befehl eine neue Verteilungskonfiguration für Ihr AMI mithilfe der erstellenAWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Dateibearbeitungstool, um eine JSON-Datei mit Schlüsseln zu erstellen, die in einem der folgenden Beispiele angezeigt werden, und Werten, die für Ihre Umgebung gültig sind. Diese Beispiele definieren, welche AWS-Konten- AWS Organizations oder Organisationseinheiten (OUs) über die Berechtigung zum Starten des AMI verfügen, das Sie an die angegebenen Regionen verteilen. Benennen Sie die Datei zur `create-ami-distribution-configuration.json`Verwendung im nächsten Schritt:

   Accounts

   In diesem Beispiel wird ein AMI an zwei Regionen verteilt und angegeben, AWS-Konten die in jeder Region über Startberechtigungen verfügen.

```
{
    "name": "MyExampleAccountDistribution",
    "description": "Copies AMI to eu-west-1, and specifies accounts that can
 launch instances in each Region.",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter
 references",
                "amiTags": {
                    "KeyName": "Some Value"
                },
                "launchPermission": {
                    "userIds": [
                        "987654321012"
                    ]
                }
            }
        },
        {
            "region": "eu-west-1",
            "amiDistributionConfiguration": {
```

```
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                "amiTags": {
                    "KeyName": "Some value"
                },
                "launchPermission": {
                    "userIds": [
                        "100000000001"
                    ]
                }
            }
        }
    ]
}
```

Organizations and OUs

In diesem Beispiel wird ein AMI an die Quellregion verteilt und die Startberechtigungen für die Organisation und Organisationseinheit angegeben.

```
{
  "name": "MyExampleAWSOrganizationDistribution",
  "description": "Shares AMI with the Organization and OU",
  "distributions": [
    {
      "region": "us-west-2",
      "amiDistributionConfiguration": {
        "name": "Name {{ imagebuilder:buildDate }}",
        "launchPermission": {
          "organizationArns": [
            "arn:aws:organizations::123456789012:organization/o-
myorganization123"
          ],
          "organizationalUnitArns": [
            "arn:aws:organizations::123456789012:ou/o-123example/ou-1234-
myorganizationalunit"
          ]
        }
      }
    }
  ]
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://create-ami-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Weitere Informationen finden Sie unter create-distribution-configuration in der AWS CLI - Befehlsreferenz.

Aktualisieren der AMI-Verteilungseinstellungen (Konsole)

Sie können Ihre AMI-Verteilungseinstellungen mithilfe der Image-Builder-Konsole ändern. Die aktualisierten Verteilungseinstellungen werden für alle automatisierten und manuellen Pipeline-Bereitstellungen in Zukunft verwendet. Die von Ihnen vorgenommenen Änderungen gelten jedoch nicht für Ressourcen, die Image Builder bereits verteilt hat. Wenn Sie beispielsweise ein AMI an eine Region verteilt haben, die Sie später aus Ihrer Verteilung entfernen, verbleibt das bereits verteilte AMI in dieser Region, bis Sie es manuell entfernen.

Aktualisieren der AMI-Verteilungskonfiguration

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Verteilungseinstellungen aus. Dies zeigt eine Liste der Verteilungskonfigurationen, die unter Ihrem Konto erstellt werden.
3. Um Details anzuzeigen oder eine Verteilungskonfiguration zu aktualisieren, wählen Sie den Link Konfigurationsname. Dadurch wird die Detailansicht für die Verteilungseinstellungen geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Konfigurationsnamen aktivieren und dann Details anzeigen auswählen.

4. Um die Verteilungskonfiguration zu bearbeiten, wählen Sie oben rechts im Abschnitt Verteilungsdetails die Option Bearbeiten aus. Einige Felder sind gesperrt, z. B. der Name der Verteilungskonfiguration und die Standardregion, die als Region 1 angezeigt wird. Weitere Informationen zu den Einstellungen für die Verteilungskonfiguration finden Sie unter Erstellen einer AMI-Verteilungskonfiguration (Konsole).
5. Wählen Sie abschließend Änderungen speichern aus.

Erstellen von Verteilungseinstellungen für ein Windows-AMI mit aktiviertem EC2 Fast Launch (AWS CLI)

Das folgende Beispiel zeigt, wie Sie mit dem create-distribution-configuration Befehl Verteilungseinstellungen erstellen, für die EC2 Fast Launch für Ihr AMI konfiguriert ist, indem Sie die verwendenAWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Dateibearbeitungstool, um eine JSON-Datei mit Schlüsseln zu erstellen, wie im folgenden Beispiel gezeigt, sowie Werte, die für Ihre Umgebung gültig sind.

   In diesem Beispiel werden Instances für alle seine Zielressourcen gleichzeitig gestartet, da die maximale Anzahl paralleler Starts größer ist als die Anzahl der Zielressourcen. Diese Datei heißt `ami-dist-config-win-fast-launch.json` im Befehlsbeispiel, das im nächsten Schritt gezeigt wird.

```
{
"name": "WinFastLaunchDistribution",
"description": "An example of Windows AMI EC2 Fast Launch settings in the
 distribution configuration.",
"distributions": [
    {
        "region": "us-west-2",
        "amiDistributionConfiguration": {
            "name": "Name {{imagebuilder:buildDate}}",
```

```
            "description": "Includes Windows AMI EC2 Fast Launch settings with
 cross-account distribution.",
            "amiTags": {
                "KeyName": "Some Value"
            }
        },
        "fastLaunchConfigurations": [{
            "enabled": true,
            "snapshotConfiguration": {
                "targetResourceCount": 5
            },
            "maxParallelLaunches": 6,
            "launchTemplate": {
                "launchTemplateId": "lt-0ab1234c56d789012",
                "launchTemplateVersion": "1"
             },
             "accountId": "123456789012"
        }],
        "launchTemplateConfigurations": [{
                    "launchTemplateId": "lt-0ab1234c56d789012",
                    "setDefaultVersion": true
          }]
    }]
}
```

> **Note**
>
> Sie können die `launchTemplateName` anstelle der `launchTemplateId` im `launchTemplate` Abschnitt angeben, aber Sie können nicht sowohl den Namen als auch die ID angeben.

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://ami-
dist-config-win-fast-launch.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.

- Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie unter create-distribution-configuration in der AWS CLI - Befehlsreferenz.

Erstellen von Verteilungseinstellungen für VM-Ausgabedatenträger (AWS CLI)

Das folgende Beispiel zeigt, wie Sie mit dem create-distribution-configuration Befehl Verteilungseinstellungen erstellen, die bei jedem Image-Build VM-Image-Datenträger nach Amazon S3 exportieren.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Sie können den create-distribution-configuration Befehl optimieren, den Sie in der verwendenAWS CLI. Erstellen Sie dazu eine JSON-Datei, die alle Exportkonfigurationen enthält, die Sie an den Befehl übergeben möchten.

   > **Note**
   >
   > Die Namenskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Anforderungsparameter der Image-Builder-API-Aktion angegeben ist. Informationen zum Überprüfen der API-Befehlsanforderungsparameter finden Sie unter dem CreateDistributionConfiguration Befehl in der EC2 Image Builder-API-Referenz . Um die Datenwerte als Befehlszeilenparameter bereitzustellen, beziehen Sie sich auf die in der -AWS CLIBefehlsreferenz angegebenen Parameternamen. auf den Befehl create-distribution-configuration als Optionen.

   Im Folgenden finden Sie eine Zusammenfassung der Parameter, die wir im `s3ExportConfiguration` JSON-Objekt für dieses Beispiel angeben:

   - roleName (Zeichenfolge, erforderlich) – Der Name der Rolle, die VM Import/Export die Berechtigung zum Exportieren von Images in Ihren S3-Bucket erteilt.

- diskImageFormat (Zeichenfolge, erforderlich) – Exportieren Sie das aktualisierte Datenträgerabbild in eines der folgenden unterstützten Formate:
  - Virtual Hard Disk (VHD) – kompatibel mit Citrix Xen und Microsoft Hyper-V-Virtualisierungsprodukten.
  - Streamoptimierte ESX Virtual Machine Disk (VMDK) – kompatibel mit VMware ESX und VMware vSphere Version 4, 5 und 6.
  - Roh – Rohformat.
- s3Bucket (Zeichenfolge, erforderlich) – Der S3-Bucket, in dem die Ausgabe-Festplattenabbilder für Ihre VM gespeichert werden sollen.

Speichern Sie die Datei als `export-vm-disks.json`. Verwenden Sie den Dateinamen im create-distribution-configuration Befehl .

```
{
    "name": "example-distribution-configuration-with-vm-export",
    "description": "example",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
             "description": "example-with-vm-export"

            },
            "s3ExportConfiguration": {
                "roleName": "vmimport",
                "diskImageFormat": "RAW",
                "s3Bucket": "vm-bucket-export"
            }
        }],
    "clientToken": "abc123def4567ab"
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://export-vm-disks.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie unter create-distribution-configuration in der AWS CLI - Befehlsreferenz.

Aktualisieren der AMI-Verteilungseinstellungen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den update-distribution-configuration Befehl verwenden, um Verteilungseinstellungen für Ihr AMI mithilfe der zu aktualisierenAWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den im folgenden Beispiel gezeigten Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen verwendet`update-ami-distribution-configuration.json`.

```
{
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-ami-distribution-
configuration.json",
    "description": "Copies AMI to eu-west-2, and specifies accounts that can launch
 instances in each Region.",
    "distributions": [
      {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter references",
                "launchPermissions": {
                    "userIds": [
```

```
                        "987654321012"
                    ]
                }
            }
        },
        {
            "region": "eu-west-2",
            "amiDistributionConfiguration": {
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                "tags": {
                    "KeyName": "Some value"
                },
                "launchPermissions": {
                    "userIds": [
                        "100000000001"
                    ]
                }
            }
        }
    ]
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder update-distribution-configuration --cli-input-json file://update-ami-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie unter update-distribution-configuration in der AWS CLI -Befehlsreferenz. Informationen zum Aktualisieren von Tags für Ihre Verteilungskonfigurationsressource finden Sie im Markieren von Ressourcen Abschnitt .

Erstellen und Aktualisieren von Verteilungseinstellungen für Container-Images

In diesem Abschnitt wird das Erstellen und Aktualisieren von Verteilungseinstellungen für Image-Builder-Container-Images behandelt.

Inhalt

Erstellen von Verteilungseinstellungen für Image-Builder-Container-Images (AWS CLI)

Mit einer Verteilungskonfiguration können Sie den Namen und die Beschreibung Ihres Ausgabe-Container-Images angeben und das Container-Image in andere AWS Regionen replizieren. Sie können auch separate Tags auf die Verteilungskonfigurationsressource und auf die Container-Images innerhalb jeder Region anwenden.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den im folgenden Beispiel gezeigten Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-container-distribution-configuration.json`:

```
{
 "name": "distribution-configuration-name",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
     {
         "region": "us-west-2",
         "containerDistributionConfiguration": {
             "description": "My test image.",
```

```
            "targetRepository": {
                "service": "ECR",
                "repositoryName": "testrepo"
            },
            "containerTags": ["west2", "image1"]
        }
    },
    {
        "region": "us-east-1",
        "containerDistributionConfiguration": {
            "description": "My test image.",
            "targetRepository": {
                "service": "ECR",
                "repositoryName": "testrepo"
            },
           "containerTags": ["east1", "imagedist"]
        }
    }
 ],
 "tags": {
    "DistributionConfigurationTestTagKey1":
 "DistributionConfigurationTestTagValue1",
    "DistributionConfigurationTestTagKey2":
 "DistributionConfigurationTestTagValue2"
 }
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://create-container-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie unter create-distribution-configuration in der AWS CLI - Befehlsreferenz.

Aktualisieren der Verteilungseinstellungen für Ihr Container-Image (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den -update-distribution-configurationBefehl verwenden, um Verteilungseinstellungen für Ihr Container-Image mithilfe der zu aktualisierenAWS CLI. Sie können auch Tags für die Container-Images innerhalb jeder Region aktualisieren.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit Schlüsseln zu erstellen, die im folgenden Beispiel gezeigt werden, sowie Werte, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`update-container-distribution-configuration.json`:

```
{
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-container-distribution-
configuration.json",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
     {
         "region": "us-west-2",
         "containerDistributionConfiguration": {
             "description": "My test image.",
             "targetRepository": {
                 "service": "ECR",
                 "repositoryName": "testrepo"
             },
             "containerTags": ["west2", "image1"]
         }
     },
     {
         "region": "us-east-2",
         "containerDistributionConfiguration": {
             "description": "My test image.",
             "targetRepository": {
                 "service": "ECR",
```

```
                "repositoryName": "testrepo"
            },
           "containerTags": ["east2", "imagedist"]
        }
    }
 ]
}
```

2. Führen Sie den folgenden Befehl mit der Datei aus, die Sie als Eingabe erstellt haben:

```
aws imagebuilder update-distribution-configuration --cli-input-json file://update-container-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Weitere Informationen finden Sie unter update-distribution-configuration in der AWS CLI -Befehlsreferenz. Informationen zum Aktualisieren von Tags für Ihre Verteilungskonfigurationsressource finden Sie im Markieren von Ressourcen Abschnitt .

Kontoübergreifende AMI-Verteilung mit Image Builder einrichten

In diesem Abschnitt wird beschrieben, wie Sie Verteilungseinstellungen konfigurieren können, um ein Image Builder-AMI für andere von Ihnen angegebene Konten bereitzustellen.

Das Zielkonto kann dann das AMI nach Bedarf starten oder ändern.

> **Note**
>
> AWS CLI -Befehlsbeispiele in diesem Abschnitt gehen davon aus, dass Sie zuvor JSON-Dateien für Image-Rezepte und Infrastrukturkonfiguration erstellt haben. Informationen zum Erstellen der JSON-Datei für ein Image-Rezept finden Sie unter Erstellen eines

Image-Rezepts mit der AWS CLI. Informationen zum Erstellen der JSON-Datei für eine Infrastrukturkonfiguration finden Sie unter Erstellen einer Infrastrukturkonfiguration.

Voraussetzungen

Um sicherzustellen, dass Zielkonten Instances erfolgreich von Ihrem Image-Builder-Image aus starten können, müssen Sie die entsprechenden Berechtigungen für alle Zielkonten in allen Regionen konfigurieren.

Wenn Sie Ihr AMI mit AWS Key Management Service (AWS KMS) verschlüsseln, müssen Sie einen AWS KMS key für Ihr Konto konfigurieren, der zum Verschlüsseln des neuen Images verwendet wird.

Wenn Image Builder eine kontoübergreifende Verteilung für verschlüsselte AMIs durchführt, wird das Image im Quellkonto entschlüsselt und an die Zielregion übertragen, wo es mit dem für diese Region angegebenen Schlüssel erneut verschlüsselt wird. Da Image Builder im Namen des Zielkontos agiert und eine IAM-Rolle verwendet, die Sie in der Zielregion erstellen, muss dieses Konto Zugriff auf Schlüssel sowohl in der Quell- als auch in der Zielregion haben.

Verschlüsselungsschlüssel

Die folgenden Voraussetzungen sind erforderlich, wenn Ihr Image mit verschlüsselt istAWS KMS. Die IAM-Voraussetzungen werden im nächsten Abschnitt behandelt.

Anforderungen an das Quellkonto

- Erstellen Sie einen KMS-Schlüssel in Ihrem Konto in allen Regionen, in denen Sie Ihr AMI erstellen und verteilen. Sie können auch einen vorhandenen Schlüssel verwenden.
- Aktualisieren Sie die Schlüsselrichtlinie für alle diese Schlüssel, damit Zielkonten Ihren Schlüssel verwenden können.

Anforderungen an das Zielkonto

- Fügen Sie eine Inline-Richtlinie zu hinzu`EC2ImageBuilderDistributionCrossAccountRole`, die es der Rolle ermöglicht, die erforderlichen Aktionen zum Verteilen eines verschlüsselten AMI auszuführen. Informationen zu den IAM-Konfigurationsschritten finden Sie im Abschnitt IAM-Richtlinien Voraussetzungen.

Weitere Informationen zum kontoübergreifenden Zugriff mit AWS KMSfinden Sie unter Benutzern in anderen Konten die Verwendung eines KMS-Schlüssels erlauben im AWS Key Management Service -Entwicklerhandbuch.

Geben Sie Ihren Verschlüsselungsschlüssel im Image-Rezept wie folgt an:

- Wenn Sie die Image-Builder-Konsole verwenden, wählen Sie Ihren Verschlüsselungsschlüssel aus der Dropdown-Liste Verschlüsselung (KMS-Alias) im Abschnitt Speicher (Volumes) Ihres Rezepts aus.
- Wenn Sie die CreateImageRecipe -API-Aktion oder den -create-image-recipeBefehl in der verwendenAWS CLI, konfigurieren Sie Ihren Schlüssel im `ebs` Abschnitt unter `blockDeviceMappings` in Ihrer JSON-Eingabe.

  Der folgende JSON-Ausschnitt zeigt Verschlüsselungseinstellungen für ein Image-Rezept. Sie müssen nicht nur Ihren Verschlüsselungsschlüssel angeben, sondern auch das `encrypted` Flag auf setzen`true`.

```
{
 ...
 "blockDeviceMappings": [
 {
  "deviceName": "Example root volume",
  "ebs": {
   "deleteOnTermination": true,
   "encrypted": true,
   "iops": 100,
   "kmsKeyId": "image-owner-key-id",
   ...
  },
  ...
 }],
 ...
}
```

IAM-Richtlinien

Gehen Sie folgendermaßen vor, um kontoübergreifende Verteilungsberechtigungen in AWS Identity and Access Management (IAM) zu konfigurieren:

1. Um Image-Builder-AMIs zu verwenden, die auf Konten verteilt sind, muss der Eigentümer des Zielkontos in seinem Konto eine neue IAM-Rolle namens erstellen`EC2ImageBuilderDistributionCrossAccountRole`.
2. Sie müssen der Rolle Ec2ImageBuilderCrossAccountDistributionAccess-Richtlinie anfügen, um die kontoübergreifende Verteilung zu aktivieren. Weitere Informationen zu verwalteten Richtlinien finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien im AWS Identity and Access Management -Benutzerhandbuch.
3. Stellen Sie sicher, dass die Quellkonto-ID der Vertrauensrichtlinie hinzugefügt wird, die der IAM-Rolle des Zielkontos zugeordnet ist. Weitere Informationen zu Vertrauensrichtlinien finden Sie unter Ressourcenbasierte Richtlinien im AWS Identity and Access Management -Benutzerhandbuch.
4. Wenn das von Ihnen verteilte AMI verschlüsselt ist, muss der Eigentümer des Zielkontos die folgende Inline-Richtlinie zu `EC2ImageBuilderDistributionCrossAccountRole` in seinem Konto hinzufügen, damit er Ihre KMS-Schlüssel verwenden kann. Der `Principal` Abschnitt enthält seine Kontonummer. Auf diese Weise kann Image Builder in seinem Namen handeln, wenn es AWS KMS zum Verschlüsseln und Entschlüsseln des AMI mit den entsprechenden Schlüsseln für jede Region verwendet.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowRoleToPerformKMSOperationsOnBehalfOfTheDestinationAccount",
   "Effect": "Allow",
   "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey",
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
   ],
   "Resource": "*"
  }
 ]
}
```

Weitere Informationen zu Inline-Richtlinien finden Sie unter Inline-Richtlinien im AWS Identity and Access Management -Benutzerhandbuch.

5. Wenn Sie verwenden, `launchTemplateConfigurations` um eine Amazon EC2-Startvorlage anzugeben, müssen Sie auch die folgende Richtlinie zu Ihrem `EC2ImageBuilderDistributionCrossAccountRole` in jedem Zielkonto hinzufügen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplateVersion",
                "ec2:ModifyLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeLaunchTemplates"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:launch-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        }
    ]
```

```
}
```

Limits für die kontoübergreifende Verteilung

Bei der Verteilung von Image-Builder-Images auf Konten gibt es einige Einschränkungen:

- Das Zielkonto ist für jede Zielregion auf 50 gleichzeitige AMI-Kopien beschränkt.
- Wenn Sie ein PV-Virtualisierungs-AMI (Paravirtual) in eine andere -Region kopieren möchten, muss die Zielregion PV-Virtualisierungs-AMIs unterstützen. Weitere Informationen finden Sie unter Linux AMI-Virtualisierungstypen.
- Sie können keine unverschlüsselte Kopie eines verschlüsselten Snapshots erstellen. Wenn Sie keinen AWS Key Management Service (AWS KMS) vom Kunden verwalteten Schlüssel für den `KmsKeyId` Parameter angeben, verwendet Image Builder den Standardschlüssel für Amazon Elastic Block Store (Amazon EBS). Weitere Informationen finden Sie unter Amazon EBS-Verschlüsselung im Amazon Elastic Compute Cloud-Benutzerhandbuch.

Weitere Informationen finden Sie unter CreateDistributionConfiguration in der API-Referenz zu EC2 Image Builder.

Kontoübergreifende Verteilung für ein Image Builder-AMI konfigurieren (Konsole)

In diesem Abschnitt wird beschrieben, wie Sie Verteilungseinstellungen für die kontoübergreifende Verteilung Ihrer Image-Builder-AMIs mithilfe der erstellen und konfigurierenAWS Management Console. Die Konfiguration der kontoübergreifenden Verteilung erfordert bestimmte IAM-Berechtigungen. Sie müssen die Voraussetzungen für diesen Abschnitt ausfüllen, bevor Sie fortfahren.

Gehen Sie folgendermaßen vor, um Verteilungseinstellungen in der Image-Builder-Konsole zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Verteilungseinstellungen aus. Dies zeigt eine Liste der Verteilungseinstellungen, die unter Ihrem Konto erstellt werden.
3. Wählen Sie oben auf der Seite Verteilungseinstellungen die Option Verteilungseinstellungen erstellen aus. Dadurch gelangen Sie zur Seite Verteilungseinstellungen erstellen.
4. Wählen Sie im Abschnitt Image-Typ die Option Amazon Machine Image (AMI) als Ausgabetyp aus. Dies ist die Standardeinstellung.

5. Geben Sie im Abschnitt Allgemein den Name nder Verteilungseinstellungen-Ressource ein, die Sie erstellen möchten (erforderlich).
6. Geben Sie im Abschnitt Regionseinstellungen eine 12-stellige Konto-ID ein, an die Sie Ihr AMI verteilen möchten, und drücken Sie die Eingabetaste für die ausgewählte Region. Dadurch wird geprüft, ob die Formatierung korrekt ist, und anschließend wird die Konto-ID angezeigt, die Sie unter dem Feld eingegeben haben. Wiederholen Sie den Vorgang, um weitere Konten hinzuzufügen.

   Um ein von Ihnen eingegebenes Konto zu entfernen, wählen Sie das X, das rechts neben der Konto-ID angezeigt wird.

   Geben Sie den Namen des Ausgabe-AMI für jede Region ein.
7. Fahren Sie mit der Angabe zusätzlicher Einstellungen fort, die Sie benötigen, und wählen Sie Einstellungen erstellen, um Ihre neue Verteilungseinstellungen-Ressource zu erstellen.

Kontoübergreifende Verteilung für ein Image Builder-AMI konfigurieren (AWS CLI)

In diesem Abschnitt wird beschrieben, wie Sie eine Datei mit Verteilungseinstellungen konfigurieren und den create-image Befehl in der verwendenAWS CLI, um ein Image-Builder-AMI zu erstellen und über -Konten zu verteilen.

Die Konfiguration der kontoübergreifenden Verteilung erfordert bestimmte IAM-Berechtigungen. Sie müssen die Voraussetzungen für diesen Abschnitt abschließen, bevor Sie den create-image Befehl ausführen.

1. Konfigurieren einer Verteilungseinstellungsdatei

   Bevor Sie den create-image Befehl in der verwenden, AWS CLI um ein Image Builder-AMI zu erstellen, das an ein anderes Konto verteilt ist, müssen Sie eine `DistributionConfiguration` JSON-Struktur erstellen, die die Zielkonto-IDs in den `AmiDistributionConfiguration` Einstellungen angibt. Sie müssen mindestens eine `AmiDistributionConfiguration` in der Quellregion angeben.

   Die folgende Beispieldatei mit dem Namen zeigt `create-distribution-configuration.json`die Konfiguration für die kontoübergreifende Image-Verteilung in der -Quellregion.

```
{
```

```
 "name": "cross-account-distribution-example",
 "description": "Cross Account Distribution Configuration Example",
 "distributions": [
  {
   "amiDistributionConfiguration": {
    "targetAccountIds": ["123456789012", "987654321098"],
    "name": "Name {{ imagebuilder:buildDate }}",
    "description": "ImageCopy Ami Copy Configuration"
   },
   "region": "us-west-2"
  }
 ]
}
```

2. Erstellen der Verteilungseinstellungen

   Um eine Image Builder-Verteilungseinstellungsressource mit dem create-distribution-configuration Befehl in der zu erstellenAWS CLI, geben Sie die folgenden Parameter im Befehl an:

   - Geben Sie den Namen der Verteilung im `--name` Parameter ein.
   - Fügen Sie die JSON-Datei der Verteilungskonfiguration an, die Sie im `--cli-input-json` Parameter erstellt haben.

   ```
   aws imagebuilder create-distribution-configuration --name my distribution name --
   cli-input-json file://create-distribution-configuration.json
   ```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Sie können JSON auch direkt im Befehl bereitstellen, indem Sie den *`--distributions`* Parameter verwenden.

Konfigurieren von AMI-Verteilungseinstellungen für die Verwendung einer Amazon EC2-Startvorlage

Um ein konsistentes Starterlebnis für Ihr Image-Builder-AMI in Zielkonten und Regionen sicherzustellen, können Sie mithilfe von eine Amazon EC2-Startvorlage in Ihren Verteilungseinstellungen angeben`launchTemplateConfigurations`. Wenn während des Verteilungsprozesses vorhanden `launchTemplateConfigurations` sind, erstellt Image Builder eine neue Version der Startvorlage, die alle ursprünglichen Einstellungen aus der Vorlage und die neue AMI-ID aus dem Build enthält. Weitere Informationen zum Starten einer EC2-Instance mithilfe einer Startvorlage finden Sie je nach Zielbetriebssystem unter einem der folgenden Links.

- Starten einer Linux-Instance über eine Startvorlage
- Starten einer Windows-Instance über eine Startvorlage

Fügen Sie Ihren AMI-Verteilungseinstellungen (Konsole) eine Amazon EC2-Startvorlage hinzu

Um eine Startvorlage mit Ihrem Ausgabe-AMI bereitzustellen, führen Sie die folgenden Schritte in der -Konsole aus:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Verteilungseinstellungen aus. Dies zeigt eine Liste der Verteilungseinstellungen, die unter Ihrem Konto erstellt werden.
3. Wählen Sie oben auf der Seite Verteilungseinstellungen die Option Verteilungseinstellungen erstellen aus. Dadurch wird die Seite Verteilungseinstellungen erstellen geöffnet.
4. Wählen Sie im Abschnitt Image type den Ausgabetyp Amazon Machine Image (AMI) aus. Dies ist die Standardeinstellung.
5. Geben Sie im Abschnitt Allgemein den Name nder Verteilungseinstellungen-Ressource ein, die Sie erstellen möchten (erforderlich).
6. Wählen Sie im Abschnitt Regionseinstellungen den Namen einer EC2-Startvorlage aus der Liste aus. Wenn Ihr Konto keine Startvorlagen enthält, wählen Sie Neue Startvorlage erstellen aus, wodurch die Startvorlagen im EC2-Dashboard geöffnet werden.

   Aktivieren Sie das Kontrollkästchen Standardversion festlegen, um die Standardversion der Startvorlage auf die neue Version zu aktualisieren, die Image Builder mit Ihrem Ausgabe-AMI erstellt.

Um der ausgewählten Region eine weitere Startvorlage hinzuzufügen, wählen Sie Konfiguration der Startvorlage hinzufügen aus.

Um eine Startvorlage zu entfernen, wählen Sie Entfernen aus.

7. Fahren Sie mit der Angabe zusätzlicher Einstellungen fort, die Sie benötigen, und wählen Sie Einstellungen erstellen, um Ihre neue Verteilungseinstellungen-Ressource zu erstellen.

Fügen Sie Ihren AMI-Verteilungseinstellungen eine Amazon EC2-Startvorlage hinzu (AWS CLI)

In diesem Abschnitt wird beschrieben, wie Sie eine Verteilungseinstellungsdatei mit einer Startvorlage konfigurieren und den create-image Befehl in der verwenden, AWS CLI um ein Image Builder-AMI und eine neue Version der Startvorlage zu erstellen und zu verteilen, die es verwendet.

1. Konfigurieren einer Verteilungseinstellungsdatei

   Bevor Sie ein Image Builder-AMI mit einer Startvorlage erstellen können, müssen AWS CLISie mithilfe der eine JSON-Struktur für die Verteilungskonfiguration erstellen, die die `launchTemplateConfigurations` Einstellungen angibt. Sie müssen mindestens einen `launchTemplateConfigurations` Eintrag in der Quellregion angeben.

   Die folgende Beispieldatei mit dem Namen `create-distribution-config-launch-template.json`zeigt einige mögliche Szenarien für die Konfiguration der Startvorlage in der Quellregion.

```
{
    "name": "NewDistributionConfiguration",
    "description": "This is just a test",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "test-{{imagebuilder:buildDate}}-
{{imagebuilder:buildVersion}}",
                "description": "description"
            },
            "launchTemplateConfigurations": [
                {
                    "launchTemplateId": "lt-0a1bcde2fgh34567",
```

```
                    "accountId": "935302948087",
                    "setDefaultVersion": true
                },
                {
                    "launchTemplateId": "lt-0aaa1bcde2ff3456"
                },
                {
                    "launchTemplateId": "lt-12345678901234567",
                    "accountId": "123456789012"
                }
            ]
        }
    ],
    "clientToken": "clientToken1"
}
```

2. Erstellen der Verteilungseinstellungen

   Um eine Image Builder-Verteilungseinstellungsressource mit dem create-distribution-configuration Befehl in der zu erstellenAWS CLI, geben Sie die folgenden Parameter im Befehl an:

   - Geben Sie den Namen der Verteilung im `--name` Parameter ein.
   - Fügen Sie die JSON-Datei der Verteilungskonfiguration an, die Sie im `--cli-input-json` Parameter erstellt haben.

   ```
   aws imagebuilder create-distribution-configuration --name my distribution name--
   cli-input-json file://create-distribution-config-launch-template.json
   ```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Sie können JSON auch direkt im Befehl bereitstellen, indem Sie den `--distributions` Parameter verwenden.

Verwalten von Lebenszyklusrichtlinien für EC2 Image Builder-Images

Wenn Sie benutzerdefinierte Images erstellen, ist es wichtig, dass Sie diese Images außer Betrieb nehmen möchten, bevor sie veraltet sind. Image-Builder-Pipelines können Updates und Sicherheitspatches automatisch anwenden. Jeder Build erstellt jedoch eine neue Version des Images und alle zugehörigen Ressourcen, die er verteilt. Frühere Versionen verbleiben in Ihrem Konto, bis Sie sie manuell löschen oder ein Skript zur Ausführung der Aufgabe erstellen.

Mit Image-Builder-Lebenszyklusverwaltungsrichtlinien können Sie den Prozess des Veraltens, Deaktivierens und Löschens veralteter Images und der zugehörigen Ressourcen automatisieren. Zu den zugeordneten Ressourcen können Ausgabe-Images gehören, die Sie an andere AWS-Konten, Organisationen und Organisationseinheiten (OUs) in verteilt habenAWS-Regionen. Sie definieren die Regeln dafür, wie und wann die einzelnen Schritte des Lebenszyklusprozesses ausgeführt werden sollen und welche Schritte in Ihre Richtlinie aufgenommen werden sollen.

Vorteile des automatisierten Lebenszyklusmanagements

Zu den allgemeinen Vorteilen des automatisierten Lebenszyklusmanagements gehören:

- Vereinfacht das Lebenszyklusmanagement für Ihre benutzerdefinierten Images mit einer automatisierten Möglichkeit, Images und zugehörige Ressourcen außer Betrieb zu nehmen.
- Hilft, Compliance-Risiken zu verhindern, die durch die Verwendung veralteter Images zum Starten neuer Instances entstehen.
- Hält Image-Inventare auf dem neuesten Stand, indem veraltete Images entfernt werden.
- Kann die Speicher- und Datenübertragungskosten senken, indem optional zugehörige Ressourcen für gelöschte Images entfernt werden.

Kosteneinsparungen erzielen

Die Verwendung von EC2 Image Builder zum Erstellen von benutzerdefinierten AMI- oder Container-Images ist kostenlos. Die Standardpreise gelten jedoch für andere -Services, die im Prozess verwendet werden. Wenn Sie ungenutzte oder veraltete Images und die zugehörigen Ressourcen

aus Ihrem entfernenAWS-Konto, können Sie auf folgende Weise Zeit- und Kosteneinsparungen erzielen:

- Reduzieren Sie die Zeit, die zum Patchen vorhandener Images benötigt wird, wenn Sie nicht auch ungenutzte oder veraltete Images patchen.
- Für AMI-Image-Ressourcen, die Sie löschen, können Sie auch verteilte AMIs und die zugehörigen Snapshots entfernen. Dieser Ansatz kann die Kosten für das Speichern von Snapshots sparen.
- Für Container-Image-Ressourcen, die Sie löschen, können Sie die zugrunde liegenden Ressourcen löschen. Dieser Ansatz kann Amazon-ECR-Speicherkosten und Datenübertragungsraten für Ihre Docker Images sparen, die in ECR-Repositorys gespeichert sind.

> **Note**
>
> Image Builder kann nicht die potenziellen Auswirkungen für alle möglichen nachgelagerten Abhängigkeiten wie Auto Scaling-Gruppen oder Startvorlagen bewerten. Sie müssen Downstream-Abhängigkeiten für Ihre Images berücksichtigen, wenn Sie Richtlinienaktionen konfigurieren.

Inhalt

Voraussetzungen für das Lebenszyklusmanagement für EC2 Image Builder-Images

Bevor Sie Richtlinien und Regeln für die Lebenszyklusverwaltung von EC2 Image Builder für Ihre Image-Ressourcen definieren können, müssen Sie die folgenden Voraussetzungen erfüllen.

- Erstellen Sie eine IAM-Rolle, die Image Builder die Berechtigung zum Ausführen von Lebenszyklusrichtlinien erteilt. Weitere Informationen zum Erstellen der Rolle finden Sie unter Erstellen einer IAM-Rolle für das Image-Builder-Lebenszyklusmanagement.
- Erstellen Sie im Zielkonto eine IAM-Rolle für die zugehörigen Ressourcen, die auf Konten verteilt wurden. Die Rolle erteilt Image Builder die Berechtigung, Lebenszyklusaktionen im Zielkonto für

zugehörige Ressourcen durchzuführen. Weitere Informationen zum Erstellen der Rolle finden Sie unter Erstellen einer IAM-Rolle für das kontoübergreifende Lebenszyklusmanagement von Image Builder.

> **Note**
>
> Diese Voraussetzung gilt nicht, wenn Sie Startberechtigungen für ein Ausgabe-AMI erteilt haben. Bei Startberechtigungen gehören dem Konto, für das Sie freigegeben haben, die Instances, die über das freigegebene AMI gestartet werden, aber alle AMI-Ressourcen verbleiben in Ihrem Konto.

- Für Container-Images müssen Sie Ihren ECR-Repositorys das folgende Tag hinzufügen, um Image Builder Zugriff zum Ausführen von Lebenszyklusaktionen auf den im Repository gespeicherten Container-Images zu gewähren: `LifecycleExecutionAccess: EC2 Image Builder`.

Erstellen einer IAM-Rolle für das Image-Builder-Lebenszyklusmanagement

Um Image Builder die Berechtigung zum Ausführen von Lebenszyklusrichtlinien zu erteilen, müssen Sie zunächst die IAM-Rolle erstellen, die es zum Ausführen von Lebenszyklusaktionen verwendet. Gehen Sie wie folgt vor, um die Servicerolle zu erstellen, die die Berechtigung erteilt.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Roles (Rollen).
3. Wählen Sie Rolle erstellen aus. Dadurch wird zum ersten Schritt des Prozesses Vertrauenswürdige Entität auswählen geöffnet, um Ihre Rolle zu erstellen.
4. Wählen Sie die Option Benutzerdefinierte Vertrauensrichtlinie für den Typ Vertrauenswürdige Entität aus.
5. Kopieren Sie die folgende JSON-Vertrauensrichtlinie und fügen Sie sie in den Textbereich Benutzerdefinierte Vertrauensrichtlinie ein, wobei Sie den Beispieltext ersetzen. Diese Vertrauensrichtlinie ermöglicht es Image Builder, die Rolle zu übernehmen, die Sie zum Ausführen von Lebenszyklusaktionen erstellen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
```

```
                "sts:AssumeRole"
            ],
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "imagebuilder.amazonaws.com"
                ]
            }
        }
    ]
}
```

6. Wählen Sie die folgende verwaltete Richtlinie aus der Liste aus: EC2ImageBuilderLifecycleExecutionPolicyund wählen Sie dann Weiter aus. Dadurch wird die Seite Name, Überprüfung und Erstellung geöffnet.

> **Tip**
>
> Filtern Sie nach `image`, um die Ergebnisse zu optimieren.

7. Geben Sie einen Role name ein.
8. Nachdem Sie Ihre Einstellungen überprüft haben, wählen Sie Rolle erstellen aus.

Erstellen einer IAM-Rolle für das kontoübergreifende Lebenszyklusmanagement von Image Builder

Um Image Builder die Berechtigung zum Ausführen von Lebenszyklusaktionen in Zielkonten für zugehörige Ressourcen zu erteilen, müssen Sie zunächst die IAM-Rolle erstellen, die es zum Ausführen von Lebenszyklusaktionen in diesen Konten verwendet. Sie müssen die Rolle im Zielkonto erstellen.

Gehen Sie wie folgt vor, um die Servicerolle zu erstellen, die die Berechtigung im Zielkonto erteilt.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Roles (Rollen).
3. Wählen Sie Rolle erstellen aus. Dadurch wird zum ersten Schritt des Prozesses Vertrauenswürdige Entität auswählen geöffnet, um Ihre Rolle zu erstellen.
4. Wählen Sie die Option Benutzerdefinierte Vertrauensrichtlinie für den Typ Vertrauenswürdige Entität aus.

5. Kopieren Sie die folgende JSON-Vertrauensrichtlinie und fügen Sie sie in den Textbereich Benutzerdefinierte Vertrauensrichtlinie ein, wobei Sie den Beispieltext ersetzen. Diese Vertrauensrichtlinie ermöglicht es Image Builder, die Rolle zu übernehmen, die Sie zum Ausführen von Lebenszyklusaktionen erstellen.

> **Note**
>
> Wenn Image Builder diese Rolle im Zielkonto verwendet, um auf zugehörige Ressourcen zu reagieren, die auf Konten verteilt wurden, handelt es im Namen des Eigentümers des Zielkontos. Das AWS-Konto, das Sie als `aws:SourceAccount` in der Vertrauensrichtlinie konfigurieren, ist das Konto, in dem Image Builder diese Ressourcen verteilt hat.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "imagebuilder.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "444455556666"
                },
                "StringLike": {
                    "aws:SourceArn": "arn:*:imagebuilder:*:*:image/*/*/*"
                }
            }
        }
    ]
}
```

6. Wählen Sie die folgende verwaltete Richtlinie aus der Liste aus: EC2ImageBuilderLifecycleExecutionPolicyund wählen Sie dann Weiter aus. Dadurch wird die Seite Name, Überprüfung und Erstellung geöffnet.

> **Tip**
>
> Filtern Sie nach `image`, um die Ergebnisse zu optimieren.

7. Geben Sie `Ec2ImageBuilderCrossAccountLifecycleAccess` als Rollennamen ein.

> **Important**
>
> `Ec2ImageBuilderCrossAccountLifecycleAccess` muss der Name dieser Rolle sein.

8. Nachdem Sie Ihre Einstellungen überprüft haben, wählen Sie Rolle erstellen aus.

Lebenszyklusverwaltungsrichtlinien für EC2 Image Builder-Image-Ressourcen

Mit Image-Lebenszyklusrichtlinien können Sie Ihre Ressourcenverwaltungsstrategie definieren, um veraltete Images und die zugehörigen Ressourcen durch einen Prozess der Veraltung, Deaktivierung und Löschung veralteter Images und der zugehörigen Ressourcen außer Betrieb zu nehmen. In diesem Abschnitt erfahren Sie, wie Sie Richtlinien auflisten, Richtliniendetails anzeigen und neue Richtlinien für AMI- und Container-Images erstellen.

Inhalt

Auflisten von Lebenszyklusverwaltungsrichtlinien für Image-Builder-Image-Ressourcen

Sie können eine Liste Ihrer Image-Lebenszyklusverwaltungsrichtlinien erhalten, die Schlüsseldetailspalten auf der Seite Lebenszyklusrichtlinienliste in der AWS Management Consoleoder mit Befehlen oder Aktionen in der Image-Builder-API, den -SDKs oder enthältAWS CLI.

Sie können eine der folgenden Methoden verwenden, um Image-Builder-Image-Lebenszyklusrichtlinienressourcen in Ihrem aufzulistenAWS-Konto. Informationen zur -API-Aktion

finden Sie unter ListLifecyclePolicies in der EC2 Image Builder-API-Referenz. Die zugehörige SDK-Anforderung finden Sie unter dem Link Siehe auch auf derselben Seite.

AWS Management Console

Die folgenden Details werden in der Konsole für Ihre vorhandenen Richtlinien angezeigt. Sie können eine beliebige Spalte auswählen, um die Sortierreihenfolge für Ihre Ergebnisse zu ändern. Die Richtlinienliste ist zunächst nach Richtliniennamen sortiert. Der Spaltenname für die aktuelle Sortierreihenfolge ist fett gedruckt.

Wenn Sie mehr als eine Seite mit Ergebnissen haben, werden die Auslagerungspfeile in der oberen rechten Ecke des Bedienfelds aktiv. Sie können Ergebnisse mit der Suchleiste nach Richtlinienname, Richtlinienstatus, Ausgabebildtyp und Image-Ressourcen-ARN filtern.

- Richtlinienname – Der Name der Richtlinie.
- Richtlinienstatus – Ob die Richtlinie aktiv oder inaktiv ist.
- Typ – Der Typ des Ausgabe-Images, das Image Builder verteilt, wenn Sie eine neue Image-Version (AMI oder Container-Image) erstellen.
- Letztes Ausführungsdatum – Das letzte Mal, dass die Lebenszyklusrichtlinie ausgeführt wurde.
- Erstellungsdatum – Der Zeitstempel ab der Erstellung der Lebenszyklusrichtlinie.
- ARN – Der Amazon-Ressourcenname (ARN) der Lebenszyklusrichtlinien-Ressource.

Gehen Sie folgendermaßen vorAWS Management Console, um Lebenszyklusrichtlinien in der aufzulisten:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Lebenszyklusrichtlinien aus. Dies zeigt eine Liste der Image-Lebenszyklusrichtlinien in Ihrem Konto.

Verfügbare Aktionen

Sie können auch die folgenden Aktionen für Ihre Lebenszyklusrichtlinie auf der Liste der Lebenszyklusrichtlinien ausführen.

Um eine neue Image-Lebenszyklusrichtlinie zu erstellen, wählen Sie Lebenszyklusrichtlinie erstellen aus. Weitere Informationen zum Erstellen einer Richtlinie finden Sie unter Erstellen von Lebenszyklusrichtlinien.

Für alle der folgenden Aktionen müssen Sie zuerst die Richtlinie auswählen. Um eine Richtlinie auszuwählen, können Sie das Kontrollkästchen neben dem Richtliniennamen aktivieren.

- Um die Richtlinie aus- oder einzuschalten, wählen Sie im Menü Aktionen die Option Richtlinie deaktivieren oder Richtlinie aktivieren aus.
- Um die Richtlinie zu ändern, wählen Sie im Menü Aktionen die Option Richtlinie bearbeiten aus.
- Um eine Richtlinie zu löschen, wählen Sie im Menü Aktionen die Option Richtlinie löschen aus.
- Um eine neue Richtlinie zu erstellen, die Ihre ausgewählte Richtlinie für Basiseinstellungen verwendet, wählen Sie im Menü Aktionen die Option Richtlinie klonen aus.

AWS CLI

Das folgende Befehlsbeispiel zeigt, wie Sie die verwenden, AWS CLI um Image-Lebenszyklusrichtlinien für eine bestimmte aufzulistenAWS-Region. Weitere Informationen zu den Parametern und Optionen, die Sie mit diesem Befehl verwenden können, finden Sie unter dem list-lifecycle-policies Befehl in der -AWS CLIBefehlsreferenz.

Beispiel:

```
aws imagebuilder list-lifecycle-policies \
--region us-west-1
```

Ausgabe:

```
{
    "lifecyclePolicySummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy1",
            "name": "sample-lifecycle-policy1",
            "status": "DISABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-11-07T14:57:01.603000-08:00",
            "tags": {}
        },
```

```
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy2",
            "name": "sample-lifecycle-policy2",
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-09-06T10:43:21.436000-07:00",
            "dateLastRun": "2023-11-13T04:43:46.106000-08:00",
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy3",
            "name": "sample-lifecycle-policy3",
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-10-19T15:16:40.046000-07:00",
            "dateUpdated": "2023-10-21T20:07:15.958000-07:00",
            "dateLastRun": "2023-11-12T09:27:45.830000-08:00"
}]}
```

> **Note**
>
> Um Ihre Standard- zu verwendenAWS-Region, führen Sie diesen Befehl ohne den Parameter aus`--region`.

Anzeigen von Details zu Lebenszyklusrichtlinien

Die Detailseite der Lebenszyklusrichtlinie in der Image-Builder-Konsole enthält einen Übersichtsabschnitt mit zusätzlichen Informationen, die in Registerkarten gruppiert sind. Die Seitenüberschrift ist der Name der Richtlinie.

Auf der Detailseite der Lebenszyklusrichtlinie in der Image-Builder-Konsole können Sie Details zu einer bestimmten Lebenszyklusrichtlinie anzeigen. Sie können auch Befehle oder Aktionen mit der Image-Builder-API, SDKs oder verwenden, AWS CLI um Richtliniendetails abzurufen.

Inhalt

Anzeigen von Details zu Lebenszyklusrichtlinien in der Image-Builder-Konsole

Die Seite mit den Image-Details in der Image-Builder-Konsole enthält einen Übersichtsabschnitt mit zusätzlichen Informationen, die in Registerkarten gruppiert sind. Die Seitenüberschrift ist der Name und die Build-Version des Rezepts, das das Image erstellt hat.

Abschnitte und Registerkarten mit Konsolendetails

- Abschnitt „Zusammenfassung“
- Registerkarte Regeln
- Registerkarte „Bereich“
- RunLog Registerkarte

Abschnitt „Zusammenfassung“

Der Übersichtsabschnitt erstreckt sich über die Breite der Seite und enthält die folgenden Details. Diese Details werden immer angezeigt.

Richtlinienstatus

Ob die Richtlinie aktiv oder inaktiv ist.

Typ

Der Typ des Ausgabe-Images, das Image Builder verteilt, wenn Sie eine neue Image-Version (AMI oder Container-Image) erstellen.

Erstellungsdatum

Der Zeitstempel ab der Erstellung der Lebenszyklusrichtlinie.

Datum der Änderung

Das letzte Mal, dass die Lebenszyklusrichtlinie aktualisiert wurde.

Datum der letzten Ausführung

Das letzte Mal, dass die Lebenszyklusrichtlinie ausgeführt wurde.

IAM role (IAM-Rolle)

Die IAM-Rolle, die Image Builder zum Ausführen von Lebenszyklusaktionen verwendet.

ARN

Der Amazon-Ressourcenname (ARN) der Lebenszyklusrichtlinien-Ressource.

Beschreibung

Die Beschreibung für die Lebenszyklusrichtlinie, falls eingegeben.

Registerkarte Regeln

Auf der Registerkarte Regeln werden die Lebenszyklusregeln angezeigt, die Sie für die Richtlinie konfiguriert haben, die Sie anzeigen. Die Registerkarte enthält die folgenden Details:

- Name – Der Name der Regel. Diese Namen sind statisch, basierend auf Richtlinienaktionen, die Sie konfigurieren können.
  - `Deprecation rule`
  - `Disable rule`
  - `Deletion rule`
- Regel – Eine kurze Beschreibung der Aktion, die für die Regel konfiguriert ist.
- Regelbedingungen – Listet die Konfiguration für die zugehörige Ressourcenbehandlung, Ausnahmen von der Regel und gegebenenfalls Aufbewahrungseinstellungen auf.

Weitere Informationen zur Regelkonfiguration finden Sie unter Funktionsweise von Lebenszyklusregeln.

Registerkarte „Bereich“

Auf der Registerkarte Umfang werden die Kriterien für die Ressourcenauswahl angezeigt, die für die Richtlinie konfiguriert sind, die Sie anzeigen. Die Registerkarte enthält die folgenden Details:

- Filter: ***`Filtertyp`*** – Der Filtertyp, den Sie zur Definition des Bereichs verwendet haben. Der Filtertyp kann einer der folgenden sein:
  - `recipes` – Die Rezepte, die zum Erstellen der Images verwendet wurden, für die die Lebenszyklusrichtlinie gilt.
  - `tags` – Ein Satz von Tags, die Image Builder verwendet, um Image-Ressourcen auszuwählen, für die die Lebenszyklusrichtlinie gilt.
- Eine Suchleiste – Sie können die Liste nach Name filtern, um die Ergebnisse zu optimieren, die auf der Registerkarte angezeigt werden.
- Name – Jede Zeile enthält einen Namen oder ein Tag, das Sie für die Filterkriterien konfiguriert haben.

- Version – Wenn Sie einen Rezeptfilter konfiguriert haben, zeigt Image Builder die Rezeptversion an.

RunLog Registerkarte

Jedes Mal, wenn Sie die Richtlinie für Ihre konfigurierten Ressourcen ausführen, speichert Image Builder Laufzeitdetails. Jede Zeile in der Tabelle stellt eine einzelne Laufzeit-Instance dar. Die Registerkarte enthält die folgenden Details:

- Ausführungs-ID – Identifiziert die Laufzeit-Instance der Lebenszyklusrichtlinie.
- Ausführungsstatus – Laufzeitstatus, der meldet, ob die Richtlinienaktion derzeit ausgeführt wird, erfolgreich ausgeführt wurde, fehlgeschlagen ist oder abgebrochen wurde.
- Betroffene Ressource – Gibt an, ob die Laufzeit-Instance Image-Ressourcen für Lebenszyklusaktionen identifiziert hat.
- Startdatum – Der Zeitstempel, zu dem die Laufzeit-Instance gestartet wurde.
- Enddatum – Der Zeitstempel, zu dem die Laufzeit-Instance beendet wurde.

Erstellen von Lebenszyklusrichtlinien

Wenn Sie eine neue Lebenszyklusrichtlinie für EC2 Image Builder erstellen, hängt die Konfiguration davon ab, für welche Art von Image die Richtlinie bestimmt ist. Die API-Aktion zum Erstellen einer Lebenszyklusrichtlinie für AMI-Image-Ressourcen und Container-Image-Ressourcen ist dieselbe (CreateLifecyclePolicy). Die Konfiguration für die Image-Ressourcen und die zugehörigen Ressourcen ist jedoch unterschiedlich. In diesem Abschnitt erfahren Sie, wie Sie Lebenszyklusverwaltungsrichtlinien für beide erstellen.

> **Note**
>
> Bevor Sie eine Lebenszyklusrichtlinie erstellen, stellen Sie sicher, dass Sie alle erfüllt habenVoraussetzungen.

Erstellen von Lebenszyklusverwaltungsrichtlinien für Image-Builder-AMI-Image-Ressourcen

Sie können eine der folgenden Methoden verwenden, um eine AMI-Image-Lebenszyklusrichtlinie mit der AWS Management Console oder zu erstellenAWS CLI. Sie können auch die

CreateLifecyclePolicy-API-Aktion verwenden. Die zugehörige SDK-Anfrage finden Sie unter dem Link Siehe auch für diesen Befehl in der EC2 Image Builder API-Referenz .

AWS Management Console

Gehen Sie folgendermaßen vorAWS Management Console, um eine Lebenszyklusrichtlinie für AMI-Image-Ressourcen in der zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Lebenszyklusrichtlinien aus.
3. Wählen Sie Lebenszyklusrichtlinie erstellen aus.
4. Konfigurieren Sie die in den folgenden Verfahren beschriebenen Richtlinieneinstellungen.
5. Um die Lebenszyklusrichtlinie zu erstellen, nachdem Sie die Einstellungen konfiguriert haben, wählen Sie Richtlinie erstellen aus.

Konfigurieren Sie die allgemeinen Einstellungen für Ihre Richtlinie.

1. Wählen Sie unter Richtlinientyp die Option AMI aus.
2. Geben Sie den Richtliniennamen ein.
3. Geben Sie optional eine Beschreibung für Ihre Lebenszyklusrichtlinie ein.
4. Standardmäßig ist Aktivieren aktiviert. Die Standardeinstellung aktiviert die Lebenszyklusrichtlinie und fügt sie sofort zum Zeitplan hinzu. Um eine Richtlinie zu erstellen, die ursprünglich deaktiviert wurde, können Sie Aktivieren deaktivieren.
5. Wählen Sie die IAM-Rolle aus, die Sie für Lebenszyklusrichtlinienberechtigungen erstellt haben. Wenn Sie diese Rolle noch nicht erstellt haben, finden Sie weitere Informationen unter Voraussetzungen .

Konfigurieren Sie den Regelbereich für Ihre Richtlinie.

In diesem Abschnitt wird die Ressourcenauswahl für Ihre Lebenszyklusrichtlinie basierend auf dem verwendeten Filtertyp konfiguriert.

1. Filtertyp: Rezepte – Um Lebenszyklusregeln auf Image-Ressourcen anzuwenden, die auf dem Rezept basieren, das sie erstellt hat, wählen Sie bis zu 50 Rezeptversionen für die Richtlinie aus.

2. Filtertyp: Tags – Um Lebenszyklusregeln auf Image-Ressourcen anzuwenden, die auf Ressourcen-Tags basieren, geben Sie eine Liste mit bis zu 50 Schlüssel-Wert-Paaren ein, mit denen die Richtlinie übereinstimmen soll.

Aktivieren Sie eine oder mehrere der folgenden Lebenszyklusregeln für , um sie auf die Ressourcen anzuwenden, die die Lebenszyklusrichtlinie auswählt. Wenn eine Ressource bei der Ausführung der Richtlinie mit mehr als einer Lebenszyklusregel übereinstimmt, führt Image Builder Regelaktionen in der folgenden Reihenfolge aus: 1) Veraltet, 2) Deaktivieren, 3) Löschen.

Veraltete Regel

Legt den Image-Builder-Image-Ressourcenstatus auf fest`Deprecated`. Image-Builder-Pipelines werden weiterhin für veraltete Images ausgeführt. Sie können optional die Veralterungszeit für zugehörige AMIs festlegen, ohne dass dies Auswirkungen auf Ihre Fähigkeit hat, neue Instances zu starten.

- Anzahl der Einheiten – Geben Sie den Ganzzahlwert für den Zeitraum an, der vergehen muss, nachdem eine Bildressource erstellt wurde, bevor sie als markiert wird`Deprecated`.
- Einheit – Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann `Days`, `WeeksMonths`, oder sein`Years`.
- Veraltete AMIs – Aktivieren Sie das Kontrollkästchen, um zugeordnete Amazon EC2-AMIs mit einem Veralterungsdatum zu markieren. Die AMIs bleiben verfügbar, und Sie können immer noch neue Instances von ihnen starten.

Regel deaktivieren

Legt den Image-Builder-Image-Ressourcenstatus auf fest`Disabled`. Dadurch wird verhindert, dass Image-Builder-Pipelines für dieses Image ausgeführt werden. Sie können optional das zugehörige AMI deaktivieren, um neue Instance-Starts zu verhindern.

- Anzahl der Einheiten – Geben Sie den Ganzzahlwert für den Zeitraum an, der vergehen muss, nachdem eine Bildressource erstellt wurde, bevor sie als markiert wird`Disabled`.
- Einheit – Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann `Days`, `WeeksMonths`, oder sein`Years`.
- AMIs deaktivieren – Aktivieren Sie das Kontrollkästchen, um die zugehörigen Amazon EC2-AMIs zu deaktivieren. Sie können die AMIs nicht mehr verwenden oder neue Instances von ihnen starten.

Regel löschen

Löscht die Image-Ressourcen nach Alter oder Anzahl. Sie definieren den Schwellenwert, der Ihren Anforderungen entspricht. Wenn eine Image-Builder-Image-Ressource den Schwellenwert überschreitet, wird sie entfernt. Sie können optional die Registrierung zugehöriger AMIs aufheben oder die Snapshots für diese AMIs löschen. Sie können auch Tags für Ressourcen angeben, die Sie über den Schwellenwert hinaus beibehalten möchten.

Wenn Sie die Regel löschen nach Alter konfigurieren, löscht Image Builder die Image-Ressource nach einem von Ihnen konfigurierten Zeitraum. Löschen Sie beispielsweise Image-Ressourcen nach 6 Monaten. Wenn Sie nach Anzahl konfigurieren, behält Image Builder die neueste Anzahl von Images bei, die Sie angeben, oder so nahe wie möglich an dieser Zahl, und löscht frühere Versionen.

- Nach Alter
  - Anzahl der Einheiten – Geben Sie den Ganzzahlwert für den Zeitraum an, der vergehen muss, nachdem eine Image-Ressource erstellt wurde, bevor sie gelöscht wird.
  - Einheit – Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann `Days`, `WeeksMonths`, oder sein`Years`.
  - Mindestens ein Image pro Rezept beibehalten – Aktivieren Sie das Kontrollkästchen, um die neueste verfügbare Image-Ressource für jede Rezeptversion beizubehalten, auf die sich diese Regel auswirkt.

  Nach Anzahl
  - Anzahl der Bilder – Geben Sie den Ganzzahlwert für die Anzahl der aktuellen Image-Ressourcen an, die für jede Rezeptversion beibehalten werden sollen.
- AMIs abmelden – Aktivieren Sie das Kontrollkästchen, um die Registrierung der zugehörigen Amazon EC2-AMIs aufzuheben. Sie können die AMIs nicht mehr verwenden oder neue Instances von ihnen starten.
- Beibehalten von Images, AMIs und Snapshots mit zugehörigen Tags – Aktivieren Sie das Kontrollkästchen, um eine Liste von Tags für Image-Ressourcen einzugeben, die Sie behalten möchten. Tags gelten für Image-Ressourcen und Amazon EC2-AMIs. Sie können bis zu 50 Schlüssel-Wert-Paare eingeben.

Markierungen (optional)

Fügen Sie Ihrer Lebenszyklusrichtlinie Tags hinzu.

AWS CLI

Um eine neue Image-Builder-Lebenszyklusrichtlinie zu erstellen, können Sie den create-lifecycle-policy Befehl in der verwendenAWS CLI.

Erstellen von Lebenszyklusverwaltungsrichtlinien für Image-Builder-Container-Image-Ressourcen

Sie können eine der folgenden Methoden verwenden, um eine Container-Image-Lebenszyklusrichtlinie mit der AWS Management Console oder zu erstellenAWS CLI. Sie können auch die CreateLifecyclePolicy -API-Aktion verwenden. Die zugehörige SDK-Anfrage finden Sie unter dem Link Siehe auch für diesen Befehl in der EC2 Image Builder API-Referenz .

AWS Management Console

Gehen Sie folgendermaßen vorAWS Management Console, um eine Lebenszyklusrichtlinie für Container-Image-Ressourcen in der zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Lebenszyklusrichtlinien aus.
3. Wählen Sie Lebenszyklusrichtlinie erstellen aus.
4. Konfigurieren Sie die in den folgenden Verfahren beschriebenen Richtlinieneinstellungen.
5. Um die Lebenszyklusrichtlinie zu erstellen, nachdem Sie die Einstellungen konfiguriert haben, wählen Sie Richtlinie erstellen aus.

Richtlinienkonfiguration: Allgemeine Einstellungen

Konfigurieren Sie die allgemeinen Einstellungen für Ihre Richtlinie.

1. Wählen Sie unter Richtlinientyp die Option AMI aus.
2. Geben Sie den Richtliniennamen ein.
3. Geben Sie optional eine Beschreibung für Ihre Lebenszyklusrichtlinie ein.
4. Standardmäßig ist Aktivieren aktiviert. Die Standardeinstellung aktiviert die Lebenszyklusrichtlinie und fügt sie sofort zum Zeitplan hinzu. Um eine Richtlinie zu erstellen, die ursprünglich deaktiviert wurde, können Sie Aktivieren deaktivieren.

5. Wählen Sie die IAM-Rolle aus, die Sie für Lebenszyklusrichtlinienberechtigungen erstellt haben. Wenn Sie diese Rolle noch nicht erstellt haben, finden Sie weitere Informationen unter Voraussetzungen .

Konfigurieren Sie den Regelbereich für Ihre Richtlinie.

In diesem Abschnitt wird die Ressourcenauswahl für Ihre Lebenszyklusrichtlinie basierend auf dem verwendeten Filtertyp konfiguriert.

1. Filtertyp: Rezepte – Um Lebenszyklusregeln auf Image-Ressourcen anzuwenden, die auf dem Rezept basieren, das sie erstellt hat, wählen Sie bis zu 50 Rezeptversionen für die Richtlinie aus.
2. Filtertyp: Tags – Um Lebenszyklusregeln auf Image-Ressourcen anzuwenden, die auf Ressourcen-Tags basieren, geben Sie eine Liste mit bis zu 50 Schlüssel-Wert-Paaren ein, mit denen die Richtlinie übereinstimmen soll.

Regel löschen

Bei Container-Images löscht diese Regel die Image Builder-Container-Image-Ressource. Sie können optional Docker-Images entfernen, die an ECR-Repositorys verteilt wurden, um zu verhindern, dass sie zum Ausführen neuer Container verwendet werden.

Wenn Sie die Regel löschen nach Alter konfigurieren, löscht Image Builder die Image-Ressource nach einem bestimmten Zeitraum, den Sie konfigurieren. Löschen Sie beispielsweise Image-Ressourcen nach 6 Monaten. Wenn Sie nach Anzahl konfigurieren, behält Image Builder die neueste Anzahl von Images bei, die Sie angeben, oder so nahe wie möglich an dieser Zahl, und löscht frühere Versionen.

- Nach Alter
  - Anzahl der Einheiten – Geben Sie den Ganzzahlwert für den Zeitraum an, der vergehen muss, nachdem eine Image-Ressource erstellt wurde, bevor sie gelöscht wird.
  - Einheit – Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann `Days`, `WeeksMonths`, oder sein`Years`.
  - Mindestens ein Image beibehalten – Aktivieren Sie das Kontrollkästchen, um nur die neueste verfügbare Image-Ressource für jede Rezeptversion beizubehalten, auf die sich diese Regel auswirkt.

Nach Anzahl

- Anzahl der Bilder – Geben Sie den Ganzzahlwert für die Anzahl der aktuellen Image-Ressourcen an, die für jede Rezeptversion beibehalten werden sollen.

- ECR-Container-Images löschen – Aktivieren Sie das Kontrollkästchen, um zugehörige Container-Images zu löschen, die in einem ECR-Repository gespeichert sind. Sie können das Container-Image nicht mehr als Basis verwenden, um neue Images zu erstellen oder neue Container auszuführen.
- Beibehalten von Bildern mit zugeordneten Tags – Aktivieren Sie das Kontrollkästchen, um eine Liste von Tags für Bildressourcen einzugeben, die Sie beibehalten möchten.

Markierungen (optional)

Fügen Sie Ihrer Lebenszyklusrichtlinie Tags hinzu.

AWS CLI

Um eine neue Image-Builder-Lebenszyklusrichtlinie zu erstellen, können Sie den create-lifecycle-policy Befehl in der verwendenAWS CLI.

Funktionsweise von Lebenszyklusverwaltungsregeln für EC2 Image Builder-Image-Ressourcen

Image-Lebenszyklusrichtlinien verwenden die Lebenszyklusregeln, die Sie definieren, um Ihre allgemeine Ressourcenmanagementstrategie zu implementieren. Die von Ihnen definierten Regeln tragen dazu bei, die Aktualität Ihrer verfügbaren Images sicherzustellen und die Kosten für die zugrunde liegende Infrastruktur wie Snapshot-Speicher für Ausgabe-AMIs oder ECR-Repository-Speicher- und Datenübertragungsraten für Container-Images zu minimieren.

Sie können die folgenden Arten von Regeln für Ihre Richtlinien konfigurieren.

Veraltete Regel

Legt den Image-Builder-Image-Ressourcenstatus auf fest`Deprecated`. Image-Builder-Pipelines werden weiterhin für veraltete Images ausgeführt. Sie können optional die Veralterungszeit für zugehörige AMIs festlegen, ohne dass dies Auswirkungen auf Ihre Fähigkeit hat, neue Instances zu starten.

Wenn ein AMI veraltet ist, wird es von allgemeinen Suchen ignoriert. Wenn Sie beispielsweise den Amazon EC2-describe-imagesBefehl in der ausführenAWS CLI, werden keine veralteten AMIs in der Ergebnismenge zurückgegeben. Sie können jedoch immer noch veraltete AMIs mit ihrer AMI-ID finden.

Diese Regel ist für Container-Images nicht verfügbar.

Regel deaktivieren

Legt den Image-Builder-Image-Ressourcenstatus auf fest`Disabled`. Dadurch wird verhindert, dass Image-Builder-Pipelines für dieses Image ausgeführt werden. Sie können optional das zugehörige AMI deaktivieren, um neue Instance-Starts zu verhindern.

Wenn ein AMI deaktiviert ist, wird es privat und kann nicht zum Starten neuer Instances verwendet werden. Wenn Sie das AMI für Konten, Organisationen oder Organisationseinheiten freigegeben haben, verlieren diese den Zugriff auf Ihr AMI, wenn es privat wird.

Diese Regel ist für Container-Images nicht verfügbar.

Regel löschen

Löscht die Image-Ressourcen nach Alter oder Anzahl. Sie definieren den Schwellenwert, der Ihren Anforderungen entspricht. Wenn eine Image-Builder-Image-Ressource den Schwellenwert überschreitet, wird sie entfernt. Sie können optional die Registrierung zugehöriger AMIs aufheben oder die Snapshots für diese AMIs löschen. Sie können auch Tags für Ressourcen angeben, die über den Schwellenwert hinaus beibehalten werden sollen.

Bei Container-Images löscht diese Regel die Image Builder-Container-Image-Ressource. Sie können optional Container-Images entfernen, die an ECR-Repositorys verteilt wurden, um zu verhindern, dass sie zum Ausführen neuer Container verwendet werden.

Inhalt

Ausschlussregeln (API/SDK/CLI)

Die folgenden Ausschlussregeln definieren Ausnahmen von den Lebenszyklusregeln für AMIs . AMIs, die die in den Ausschlussregeln angegebenen Kriterien erfüllen, werden von Lebenszyklusaktionen ausgeschlossen. Ausschlussregeln sind in der nicht verfügbarAWS Management Console.

Die folgenden Begriffe verwenden die API-Notation aus dem -`LifecyclePolicyDetailExclusionRules`Datentyp.

Ausschlussregeln

amis

Enthält die Einstellungen in , die in der folgenden Liste `LifecyclePolicyDetailExclusionRulesAmis` angezeigt werden.

tagMap

Sie können eine Liste mit bis zu 50 Tags bereitstellen, die Lebenszyklusaktionen für jede Art von Ressource überspringen.

Die folgenden Begriffe verwenden die API-Notation aus dem -`LifecyclePolicyDetailExclusionRulesAmis`Datentyp.

AMI-Ausschlussregeln

isPublic

Konfiguriert, ob öffentliche AMIs von der Lebenszyklusaktion ausgeschlossen werden.

lastLaunched

Gibt Konfigurationsdetails für Image Builder an, um die neuesten Ressourcen von Lebenszyklusaktionen auszuschließen.

Regionen

Konfiguriert AWS-Regionen , die von der Lebenszyklusaktion ausgeschlossen sind.

sharedAccounts

Gibt anAWS-Konten, dessen Ressourcen von der Lebenszyklusaktion ausgeschlossen sind.

tagMap

Listet Tags auf, die von Lebenszyklusaktionen für die AMIs ausgeschlossen werden sollen, die sie haben.

Anzeigen von Details zu Lebenszyklusverwaltungsregeln für eine Richtlinie

Regeln werden in den Lebenszyklusverwaltungsrichtlinien definiert, die Sie für Ihre Image-Builder-Image-Ressourcen erstellen. In der Konsole enthält die Seite mit den Details zur Lebenszyklusrichtlinie eine Registerkarte Regeln, die die Details der Regeln anzeigt, die Sie für die Richtlinie konfiguriert haben.

Um Richtliniendetails in der abzurufenAWS CLI, können Sie den Befehl ausführenget-lifecycle-policy. Die Richtliniendetails in der Antwort enthalten eine Liste der Aktionen (Regeln), die Sie für die Richtlinie definiert haben, die alle konfigurierten Einstellungen enthalten.

Verwalten von Build- und Test-Workflows für EC2-Image-Builder-Images

Ein Image-Workflow definiert die Reihenfolge der Schritte, die EC2 Image Builder während der Build- und Testphase des Image-Erstellungsprozesses ausführt. Dies ist Teil des gesamten Image-Builder-Workflow-Frameworks.

Image-Workflow-Vorteile

- Mit Image-Workflows haben Sie mehr Flexibilität, Transparenz und Kontrolle über den Prozess der Image-Erstellung.
- Sie können benutzerdefinierte Workflow-Schritte hinzufügen, wenn Sie Ihr Workflow-Dokument definieren, oder Sie können den Standard-Workflow von Image Builder verwenden.
- Sie können Workflow-Schritte ausschließen, die in Standard-Image-Workflows enthalten sind.
- Sie können nur Test-Workflows erstellen, die den Build-Prozess vollständig überspringen. Sie können dasselbe tun, um reine Build-Workflows zu erstellen.

> **Note**
>
> Sie können einen vorhandenen Workflow nicht ändern, aber Sie können ihn klonen oder eine neue Version erstellen.

Workflow-Framework: Stufen

Um Image-Workflows anzupassen, ist es wichtig, die Workflow-Phasen zu verstehen, aus denen das Image-Erstellungs-Workflow-Framework besteht.

Das Workflow-Framework für die Image-Erstellung umfasst die folgenden zwei verschiedenen Phasen.

1. Build-Phase (Pre-Snapshot) – Während der Build-Phase nehmen Sie Änderungen an der Amazon EC2-Build-Instance vor, auf der Ihr Basis-Image ausgeführt wird, um die Basislinie für Ihr neues Image zu erstellen. Ihr Rezept kann beispielsweise Komponenten enthalten, die eine Anwendung installieren oder die Firewall-Einstellungen des Betriebssystems ändern.

   Nach erfolgreichem Abschluss dieser Phase erstellt Image Builder einen Snapshot oder ein Container-Image, das es für die Testphase und darüber hinaus verwendet.
2. Testphase (nach dem Snapshot) – Während der Testphase gibt es einige Unterschiede zwischen Bildern, die AMIs und Container-Imageserstellen. Für AMI-Workflows startet Image Builder eine EC2-Instance aus dem Snapshot, den es als letzten Schritt der Build-Phase erstellt hat. Tests werden auf der neuen Instance ausgeführt, um Einstellungen zu validieren und sicherzustellen, dass die Instance wie erwartet funktioniert. Bei Container-Workflows werden die Tests auf derselben Instance ausgeführt, die für die Erstellung verwendet wurde.

Das Workflow-Framework umfasst auch eine Verteilungsphase. Image Builder übernimmt jedoch die Workflows für diese Phase.

Zugriff auf Services

Um Image-Workflows auszuführen, benötigt Image Builder die Berechtigung zum Ausführen von Workflow-Aktionen. Sie können die AWSServiceRoleForImageBuilder serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle für den Servicezugriff wie folgt angeben.

- Konsole – Wählen Sie im Pipeline-Assistenten Schritt 3 Definieren des Image-Erstellungsprozesses die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle aus der Liste der IAM-Rollen im Service-Zugriffsbereich aus.
- Image-Builder-API – Geben Sie in der CreateImage Aktionsanforderung die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle als Wert für den `executionRole` Parameter an.

Weitere Informationen zum Erstellen einer Servicerolle finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen -AWSService im AWS Identity and Access Management - Benutzerhandbuch.

Inhalt

Auflisten von Image-Workflows

Auf der Liste der Image-Workflows in der Image-Builder-Konsole können Sie eine Liste der Image-Workflow-Ressourcen abrufen, die Sie besitzen oder auf die Sie Zugriff haben, sowie einige wichtige Details zu diesen Ressourcen. Sie können auch Befehle oder Aktionen mit der Image-Builder-API, SDKs oder verwenden, AWS CLI um Image-Workflows in Ihrem Konto aufzulisten.

Sie können eine der folgenden Methoden verwenden, um Image-Workflow-Ressourcen aufzulisten, die Sie besitzen oder auf die Sie Zugriff haben. Informationen zur -API-Aktion finden Sie unter ListWorkflows in der EC2 Image Builder-API-Referenz . Die zugehörige SDK-Anforderung finden Sie unter dem Link Siehe auch auf derselben Seite.

Console

Workflow-Details

Zu den Details auf der Liste der Image-Workflows in der Image-Builder-Konsole gehören die folgenden:

- Workflow – Der Name der neuesten Version der Image-Workflow-Ressource. In der Image-Builder-Konsole verweist die Spalte Workflow auf die Workflow-Detailseite.
- Version – Die neueste Version der Image-Workflow-Ressource.
- Typ – Der Workflow-Typ: `BUILD` oder `TEST`.
- Eigentümer – Der Eigentümer der Workflow-Ressource.
- Erstellungszeit – Das Datum und die Uhrzeit, zu der Image Builder die neueste Version der Image-Workflow-Ressource erstellt hat.
- ARN – Der Amazon-Ressourcenname (ARN) der aktuellen Version der Image-Workflow-Ressource.

Auflisten von Image-Workflows

Führen Sie die folgenden Schritte aus, um Image-Workflow-Ressourcen in der Image-Builder-Konsole aufzulisten:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Workflows aus.

Filterergebnisse

Auf der Liste der Image-Workflows können Sie nach bestimmten Image-Workflows suchen, um Ihre Ergebnisse zu filtern. Die folgenden Filter sind für Image-Workflows verfügbar:

`Workflow`

Sie können einen Workflow-Namen ganz oder teilweise eingeben, um die Ergebnisse zu optimieren. Standardmäßig werden alle Workflows in der Liste angezeigt.

`Version`

Sie können eine Versionsnummer ganz oder teilweise eingeben, um die Ergebnisse zu optimieren. Standardmäßig werden alle Versionen in der Liste angezeigt.

`Type`

Sie können nach dem Workflow-Typ filtern oder alle Typen anzeigen. Standardmäßig werden alle Workflow-Typen in der Liste angezeigt.

- BUILD
- TEST

`Owner`

Wenn Sie den Besitzerfilter aus der Suchleiste auswählen, zeigt Image Builder eine Liste der Besitzer für die Image-Workflows in Ihrem Konto an. Sie können einen Besitzer aus der Liste auswählen, um die Ergebnisse zu optimieren. Standardmäßig werden alle Besitzer in der Liste angezeigt.

- AWS-Konto – Das Konto, dem die Workflow-Ressource gehört.
- Amazon – Workflow-Ressourcen, die Amazon besitzt und verwaltet.

AWS CLI

Wenn Sie den list-workflows Befehl in der ausführenAWS CLI, erhalten Sie eine Liste der Image-Workflows, die Sie besitzen oder auf die Sie Zugriff haben.

Das folgende Befehlsbeispiel zeigt, wie Sie den list-workflows Befehl ohne Filter verwenden, um alle Image-Builder-Image-Workflow-Ressourcen aufzulisten, die Sie besitzen oder auf die Sie Zugriff haben.

Beispiel: Auflisten aller Image-Workflows

```
aws imagebuilder list-workflows
```

Ausgabe:

```
{
 "workflowVersionList": [
  {
    "name": "example-test-workflow",
    "dateCreated": "2023-11-21T22:53:14.347Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "TEST",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/test/example-test-
workflow/1.0.0"
  },
  {
    "name": "example-build-workflow",
    "dateCreated": "2023-11-20T12:26:10.425Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "BUILD",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/example-
build-workflow/1.0.0"
  }
 ]
}
```

Wenn Sie den list-workflows Befehl ausführen, können Sie Filter anwenden, um die Ergebnisse zu optimieren, wie das folgende Beispiel zeigt. Weitere Informationen zum Filtern Ihrer Ergebnisse finden Sie im Befehl list-workflows in der AWS CLI -Befehlsreferenz.

Beispiel: Filtern nach Build-Workflows

```
aws imagebuilder list-workflows --filters name="type",values="BUILD"
```

Ausgabe:

```
{
 "workflowVersionList": [
  {
    "name": "example-build-workflow",
    "dateCreated": "2023-11-20T12:26:10.425Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "BUILD",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/example-
build-workflow/1.0.0"
  }
 ]
}
```

Erstellen eines Image-Workflows

Wenn Sie einen Image-Workflow erstellen, haben Sie mehr Kontrolle über Ihren Image-Erstellungsprozess. Sie können angeben, welcher Workflow ausgeführt wird, wenn Image Builder Ihr Image erstellt, und welche Workflows ausgeführt werden, wenn es das Image testet. Sie können auch einen vom Kunden verwalteten Schlüssel angeben, um Ihre Workflow-Ressourcen zu verschlüsseln. Weitere Informationen zur Verschlüsselung für Ihre Workflow-Ressourcen finden Sie unter Verschlüsselung und Schlüsselverwaltung in EC2 Image Builder.

Für die Image-Erstellung können Sie einen Build-Stufen-Workflow und einen oder mehrere Teststufen-Workflows angeben. Sie können die Build- oder Testphase je nach Bedarf sogar vollständig überspringen. Sie konfigurieren die Aktionen, die Ihr Workflow ausführt, im YAML-Definitionsdokument, das Ihr Workflow verwendet. Weitere Informationen zur Syntax für Ihr YAML-Dokument finden Sie unter Erstellen eines YAML-Workflow-Dokuments.

Wählen Sie für Schritte zum Erstellen eines neuen Build- oder Test-Workflows die Registerkarte aus, die der Umgebung entspricht, die Sie verwenden möchten.

AWS Management Console

Sie können den folgenden Prozess verwenden, um einen neuen Workflow in der Image-Builder-Konsole zu erstellen.

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Workflows aus. Dadurch wird eine Liste von Image-Workflows angezeigt, die Ihrem Konto gehören oder auf die Ihr Konto Zugriff hat.

   > **Note**
   >
   > Die von Amazon verwalteten Workflow-Ressourcen, die Image Builder für seine Standardworkflows verwendet, werden immer in Ihrer Liste angezeigt. Um Details zu diesen Workflows anzuzeigen, können Sie den Workflow-Link auswählen.

3. Um einen neuen Workflow zu erstellen, wählen Sie Image-Workflow erstellen aus. Dadurch wird die Seite Image-Workflow erstellen angezeigt.
4. Konfigurieren Sie die Details für Ihren neuen Workflow. Um einen Build-Workflow zu erstellen, wählen Sie oben im Formular die Option Erstellen aus. Um einen Test-Workflow zu erstellen, wählen Sie oben im Formular die Option Test aus. Image Builder füllt die Liste Vorlagen basierend auf dieser Option aus. Alle anderen Schritte sind für Build- und Test-Workflows identisch.

   Allgemeines

   Der allgemeine Abschnitt enthält Einstellungen, die für Ihre Workflow-Ressource gelten, wie Name und Beschreibung. Die allgemeinen Einstellungen umfassen Folgendes:

   - Name des Image-Workflows (erforderlich) – Der Name für Ihren Image-Workflow. Der Name muss in Ihrem Konto eindeutig sein. Der Name kann bis zu 128 Zeichen lang sein. Gültige Zeichen sind Buchstaben, Zahlen, Leerzeichen`-`, und `_`.
   - Version (erforderlich) – Die semantische Version für die zu erstellende Workflow-Ressource (major.minor.patch).
   - Beschreibung (optional) – Fügen Sie optional eine Beschreibung für Ihren Workflow hinzu.
   - KMS-Schlüssel (optional) – Sie können Ihre Workflow-Ressourcen mit einem vom Kunden verwalteten Schlüssel verschlüsseln. Weitere Informationen finden Sie unter Verschlüsseln von Image-Workflows mit einem vom Kunden verwalteten Schlüssel.

Definitionsdokument

Das YAML-Workflow-Dokument enthält die gesamte Konfiguration für Ihren Workflow.

Erste Schritte

- Um mit einer Image-Builder-Standardvorlage als Grundlage für Ihren Workflow zu beginnen, wählen Sie die Option Aus Vorlagen starten aus. Diese Option ist standardmäßig ausgewählt. Nachdem Sie aus der Liste Vorlagen ausgewählt haben, welche Vorlage verwendet werden soll, kopiert dies die Standardkonfiguration aus der ausgewählten Vorlage in den Inhalt für Ihr neues Workflow-Dokument, wo Sie Änderungen vornehmen können.
- Um Ihr Workflow-Dokument von Grund auf neu zu definieren, wählen Sie die Option Von Grund auf neu starten aus. Dadurch werden die Inhalte mit einer kurzen Übersicht über einige wichtige Teile des Dokumentformats gefüllt, um Ihnen den Einstieg zu erleichtern.

Das Inhaltsfenster enthält unten eine Statusleiste, in der Warnungen oder Fehler für Ihr YAML-Dokument angezeigt werden. Weitere Informationen zum Erstellen eines YAML-Workflow-Dokuments finden Sie unter Erstellen eines YAML-Workflow-Dokuments.

5. Wenn Sie Ihren Workflow abgeschlossen haben oder den Fortschritt speichern und später darauf zurückkommen möchten, wählen Sie Workflow erstellen aus.

AWS CLI

Bevor Sie den create-workflow Befehl in der ausführenAWS CLI, müssen Sie das YAML-Dokument erstellen, das die gesamte Konfiguration für Ihren Workflow enthält. Weitere Informationen finden Sie unter Erstellen eines YAML-Workflow-Dokuments.

Das folgende Beispiel zeigt, wie Sie einen Build-Workflow mit dem AWS CLI Befehl create-workflow erstellen. Der `--data` Parameter bezieht sich auf ein YAML-Dokument, das die Build-Konfiguration für den von Ihnen erstellten Workflow enthält.

Beispiel: Workflow erstellen

```
aws imagebuilder create-workflow --name example-build-workflow --semantic-
version 1.0.0 --type BUILD --data file://example-build-workflow.yml
```

Ausgabe:

```
{
"workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
build/example-build-workflow/1.0.0/1",
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

Das folgende Beispiel zeigt, wie Sie einen Test-Workflow mit dem AWS CLI Befehl create-workflow erstellen. Der `--data` Parameter bezieht sich auf ein YAML-Dokument, das die Build-Konfiguration für den von Ihnen erstellten Workflow enthält.

Beispiel: Erstellen eines Test-Workflows

```
aws imagebuilder create-workflow --name example-test-workflow --semantic-
version 1.0.0 --type TEST --data file://example-test-workflow.yml
```

Ausgabe:

```
{
"workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
test/example-test-workflow/1.0.0/1",
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

Erstellen eines YAML-Workflow-Dokuments

Das YAML-Formatdefinitionsdokument konfiguriert Eingabe-, Ausgabe- und Workflow-Schritte für die Build- und Testphasen des Image-Build-Prozesses. Sie können mit Vorlagen beginnen, die standardisierte Schritte enthalten, oder Sie können von Grund auf neu beginnen, um Ihren eigenen Workflow zu definieren. Unabhängig davon, ob Sie eine Vorlage verwenden oder von Grund auf neu beginnen, können Sie den Workflow an Ihre Anforderungen anpassen.

Struktur eines YAML-Workflow-Dokuments

Das YAML-Workflow-Dokument, das Image Builder zum Ausführen von Image-Build- und Testaktionen verwendet, ist wie folgt strukturiert.

- Identifizierung

- Eingabeparameter
- Schritte
- Outputs

Identifizierung

Identifiziert den Workflow eindeutig. Dieser Abschnitt kann die folgenden Attribute enthalten.

Feld	Beschreibung	Typ	Erforderlich
Name	Der Name des Workflow-Dokuments.	String	Nein
description	Die Dokumentb eschreibung.	String	Nein
schemaVersion	Die Dokuments chemaversion, derzeit 1.0.	String	Ja

Beispiel

```
---
name: sample-test-image
description: Workflow for a sample image, with extra configuration options exposed
 through workflow parameters.
schemaVersion: 1.0
```

Eingabeparameter

Dieser Teil des Workflow-Dokuments definiert Eingabeparameter, die der Aufrufer angeben kann. Wenn Sie keine Parameter haben, können Sie diesen Abschnitt weglassen. Wenn Sie Parameter angeben, kann jeder Parameter die folgenden Attribute enthalten.

Feld	Beschreibung	Typ	Erforderlich	Beschränkungen
Name	Der Name des Parameters.	String	Ja	
description	Die Parameter beschreibung.	String	Nein	
default	Der Standardw ert des Parameters, wenn kein Wert angegeben wird. Wenn Sie keinen Standardwert in die Parameter definition aufnehmen, ist der Parameter wert zur Laufzeit erforderlich.	Entspricht dem Parameter datentyp.	Nein	
Typ	Der Datentyp des Parameter s. Wenn Sie den Datentyp nicht in die Parameter definition aufnehmen, wird standardmäßig ein Zeichenfo lgenwert verwendet, der	String	Ja	Der Datentyp des Parameters muss einer der folgenden sein: • `string` • `integer` • `boolean` • `stringList`

Feld	Beschreibung	Typ	Erforderlich	Beschränkungen
	zur Laufzeit erforderlich ist.			

Beispiel

Geben Sie den Parameter im Workflow-Dokument an.

```
parameters:
  - name: waitForActionAtEnd
    type: boolean
    default: true
    description: "Wait for an external action at the end of the workflow"
```

Verwenden Sie den Parameterwert im Workflow-Dokument.

```
$.parameters.waitForActionAtEnd
```

Schritte

Gibt bis zu 15 Schrittaktionen für den Workflow an. Schritte werden in der Reihenfolge ausgeführt, in der sie im Workflow-Dokument definiert sind. Im Falle eines Ausfalls wird ein Rollback in umgekehrter Reihenfolge ausgeführt, beginnend mit dem fehlgeschlagenen Schritt und umgekehrt durch vorherige Schritte.

Jeder Schritt kann sich auf die Ausgabe aller vorherigen Schrittaktionen beziehen. Dies wird als Verketten oder Verweisen auf bezeichnet. Um auf die Ausgabe einer vorherigen Schrittaktion zu verweisen, können Sie einen JSONPath-Selektor verwenden. Beispielsweise:

```
$.stepOutputs.step-name.output-name
```

Weitere Informationen finden Sie unter Verwenden dynamischer Variablen in Ihrem Workflow-Dokument.

> **Note**
>
> Obwohl der Schritt selbst kein Ausgabeattribut hat, ist jede Ausgabe einer Schrittaktion `stepOutput` für den Schritt in enthalten.

Jeder Schritt kann die folgenden Attribute enthalten.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
action	Die Workflow-Aktion, die dieser Schritt ausführt.	String	Ja		Muss eine unterstützte Schrittaktion für Image-Builder-Workflow-Dokumente sein.
`if`, gefolgt von einer Reihe von bedingten Anweisungen, die den `if` Operator ändern.	Bedingte Anweisungen fügen dem Hauptteil Ihrer Workflow-Schritte den Fluss der Kontrollentscheidungspunkte hinzu.	Diktat	Nein		Image Builder unterstützt die folgenden bedingten Anweisungen als Modifikatoren für den `if` Operator: • Verzweigungsbedingungen und Modifikatoren: `if`, `and`, `or`, `not`.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
					Verzweigungsbedingungen werden in einer Zeile selbst angegeben. • Vergleichsoperatoren: `booleanEquals`, `numberEquals`, `numberGreaterThan` `numberGreaterThanEquals`,, `numberLessThan`, `numberLessThanEquals`, `stringEquals`.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
description	Die Schrittbeschreibung.	String	Nein		Leere Zeichenfolgen sind nicht zulässig. Falls enthalten, muss die Länge 1–1024 Zeichen betragen.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
inputs	Enthält Parameter, die die Schrittaktion zum Ausführen benötigt. Sie können Schlüsselwerte als statische Werte oder mit einer JSONPath-Variable angeben, die in den richtigen Datentyp aufgelöst wird.	Diktat	Ja		

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
Name	Der Name des Schritts. Dieser Name muss innerhalb des Workflow-Dokuments eindeutig sein.	String	Ja		Die Länge muss zwischen 3 und 128 Zeichen liegen. Kann alphanumerische Zeichen und enthalten _. Keine Leerzeichen.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
onFailure	Konfiguriert die Aktion, die ausgeführt werden soll, wenn der Schritt fehlschlägt, wie folgt. Behavior • `Abort` – Fehlschlägt den Schritt fehl, der Workflow schlägt fehl und führt nach dem fehlgeschlagenen Schritt keine verbleibenden Schritte aus. Wenn das Rollback aktiviert ist, beginnt das Rollback	String	Nein	`Abort`	`Abort` | `Continue`

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
	mit dem fehlgeschlagenen Schritt und fährt fort, bis alle Schritte, die es zulassen, zurückgesetzt werden. • `Continue` – Schlägt den Schritt fehl, führt aber nach dem fehlgeschlagenen Schritt weiterhin die verbleibenden Schritte aus. In diesem Fall gibt es kein Rollback.				

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
rollbackEnabled	Konfiguriert, ob der Schritt zurückgesetzt wird, wenn ein Fehler auftritt. Sie können einen statischen booleschen Wert oder eine dynamische JSONPath-Variable verwenden, die in einen booleschen Wert aufgelöst wird.	Boolesch	Nein	`true`	`true` | `false` | oder eine JSONPath-Variable, die in „true“ oder „false“ aufgelöst wird.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
timeoutSeconds	Die maximale Zeit in Sekunden, die der Schritt vor dem Fehlschlagen und Wiederholen ausführt, wenn Wiederholungsversuche gelten.	Ganzzahl	Nein	Hängt ggf. von der Standardeinstellung ab, die für die Schrittaktion definiert ist.	Zwischen 1 und 86 400 Sekunden (maximal 24 Stunden)

Beispiel

```
steps:
  - name: LaunchTestInstance
    action: LaunchInstance
    onFailure: Abort
    inputs:
      waitFor: "ssmAgent"

  - name: ApplyTestComponents
    action: ExecuteComponents
    onFailure: Abort
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"

  - name: TerminateTestInstance
    action: TerminateInstance
    onFailure: Continue
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"
```

```
  - name: WaitForActionAtEnd
    action: WaitForAction
    if:
      booleanEquals: true
      value: "$.parameters.waitForActionAtEnd"
```

Outputs

Definiert Ausgaben für den Workflow. Jede Ausgabe ist ein Schlüssel-Wert-Paar, das den Namen der Ausgabe und den Wert angibt. Sie können Ausgaben verwenden, um Daten zur Laufzeit zu exportieren, die nachfolgende Workflows verwenden können. Dieser Abschnitt ist optional.

Jede Ausgabe, die Sie definieren, enthält die folgenden Attribute.

Feld	Beschreibung	Typ	Erforderlich
Name	Der Name der Ausgabe. Der Name muss in den Workflows, die Sie in Ihre Pipeline aufnehmen, eindeutig sein.	String	Ja
Wert	Der Wert für die Ausgabe. Der Wert der Zeichenfolge kann eine dynamische Variable sein, z. B. eine Ausgabedatei aus einer Schrittaktion. Weitere Informationen finden Sie unter Verwenden dynamischer	String	Ja

Feld	Beschreibung	Typ	Erforderlich
	Variablen in Ihrem Workflow-Dokument.		

Beispiel

Erstellen Sie eine Ausgabe-Image-ID für das Workflow-Dokument mit der Schrittausgabe aus dem `createProdImage` Schritt.

```
outputs:
  - name: 'outputImageId'
    value: '$.stepOutputs.createProdImage.imageId'
```

Weitere Informationen finden Sie in der Workflow-Ausgabe im nächsten Workflow.

```
$.workflowOutputs.outputImageId
```

Unterstützte Schrittaktionen für Ihr Workflow-Dokument

Dieser Abschnitt enthält Details zu den Schrittaktionen, die Image Builder unterstützt.

In diesem Abschnitt verwendete Begriffe

AMI

Amazon Machine Image

ARN

Amazon-Ressourcenname

Unterstützte Aktionen

- BootstrapInstanceForContainer
- CollectImageMetadata
- CollectImageScanFindings
- CreateImage

- ExecuteComponents
- LaunchInstance
- RunCommand
- RunSysPrep
- SanitizeInstance
- TerminateInstance
- WaitForAction

BootstrapInstanceForContainer

Diese Schrittaktion führt ein Serviceskript aus, um die Instance mit Mindestanforderungen zum Ausführen von Container-Workflows zu booten. Image Builder verwendet die sendCommand in der Systems Manager API, um dieses Skript auszuführen. Weitere Informationen finden Sie unter AWS Systems Manager Run Command .

> **Note**
>
> Das Bootstrap-Skript installiert die Docker-Pakete AWS CLI und , die Voraussetzungen für Image Builder sind, um Docker-Container erfolgreich zu erstellen. Wenn Sie diese Schrittaktion nicht einbeziehen, könnte der Image-Build fehlschlagen.

Standard-Timeout: 60 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der zu bootstrapenden Instance.	String	Ja		Dies muss die Ausgabe-Instance-ID aus dem Workflow-Schritt sein,

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
					der die Instance für diesen Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagerssendCommand, der das Bootstrap-Skript auf der Instance ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Vom Systems Manager zurückgegebene AusgabesendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: ContainerBootstrapStep
  action: BootstrapInstanceForContainer
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.ContainerBootstrapStep.status
```

CollectImageMetadata

Diese Schrittaktion ist nur für Build-Workflows gültig.

EC2 Image Builder führt AWS Systems Manager (Systems Manager) Agent auf den EC2-Instances aus, die es startet, um Ihr Image zu erstellen und zu testen. Image Builder sammelt zusätzliche Informationen über die Instance, die während der Erstellungsphase mit Systems Manager Inventory verwendet wurde. Zu diesen Informationen gehören der Name und die Version des Betriebssystems (OS) sowie die Liste der Pakete und der jeweiligen Versionen, wie von Ihrem Betriebssystem gemeldet.

> **Note**
>
> Diese Schrittaktion funktioniert nur für Images, die AMIs erstellen.

Standard-Timeout: 30 Minuten

Rollback: Image Builder setzt alle Systems Manager-Ressourcen zurück, die in diesem Schritt erstellt wurden.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die Build-Instance, auf die die Metadateneinstellungen angewendet werden sollen.	String	Ja		Dies muss die Ausgabe-Instance-ID aus dem Workflow-Schritt sein, der die Build-Instance für diesen

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
					Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
osVersion	Der Name und die Version des Betriebssystems, die von der Build-Instance erfasst wurden.	String
associationId	Die Systems Manager-Zuordnungs-ID, die für die Bestandserfassung verwendet wird.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: CollectMetadataStep
  action: CollectImageMetadata
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe der Schrittaktion im Workflow-Dokument.

```
$.stepOutputs.CollectMetadataStep.osVersion
```

CollectImageScanFindings

Wenn Amazon Inspector für Ihr Konto und Image-Scan für Ihre Pipeline aktiviert ist, erfasst diese Schrittaktion Image-Scan-Ergebnisse, die von Amazon Inspector für Ihre Test-Instance gemeldet wurden. Diese Schrittaktion ist für Build-Workflows nicht verfügbar.

Standard-Timeout: 120 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID für die Instance, auf der das Scannen ausgeführt wurde.	String	Ja		Dies muss die Ausgabe-Instance-ID aus dem Workflow-Schritt sein, der die Instance für diesen Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagerssendCommand, der das Skript ausgeführt hat, um Ergebnisse zu sammeln.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Vom Systems Manager zurückgegebene AusgabesendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: CollectFindingsStep
  action: CollectImageScanFindings
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.CollectFindingsStep.status
```

CreateImage

Diese Schrittaktion erstellt ein Image aus einer laufenden Instance mit der Amazon EC2`CreateImage`-API. Während des Erstellungsprozesses wartet die Schrittaktion nach Bedarf, um zu überprüfen, ob die Ressourcen den richtigen Status erreicht haben, bevor sie fortgesetzt wird.

Standard-Timeout: 720 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die Instance, aus der das neue Image erstellt werden soll.	String	Ja		Die Instance für die bereitgestellte Instance-ID muss sich beim Start dieses Schritts im `running`

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
					Status befinden.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
imageId	Die AMI-ID des erstellten Images.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: CreateImageFromInstance
  action: CreateImage
  onFailure: Abort
  inputs:
      instanceId.$: "i-1234567890abcdef0"
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.CreateImageFromInstance.imageId
```

ExecuteComponents

Diese Schrittaktion führt Komponenten aus, die im Rezept für das aktuelle Image angegeben sind, das erstellt wird. Build-Workflows führen Build-Komponenten auf der Build-Instance aus. Testworkflows führen nur Testkomponenten auf der Test-Instance aus.

Image Builder verwendet die sendCommand in der Systems Manager API, um Komponenten auszuführen. Weitere Informationen finden Sie unter AWS Systems Manager Run Command .

Standard-Timeout: 720 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID für die Instance, auf der die Komponenten ausgeführt werden sollen.	String	Ja		Dies muss die Ausgabe-Instance-ID aus dem Workflow-Schritt sein, der die Instance für diesen Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagerssendCommand, der die Komponenten auf der Instance ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Vom Systems Manager zurückgegebene AusgabesendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: ExecComponentsStep
  action: ExecuteComponents
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe der Schrittaktion im Workflow-Dokument.

```
$.stepOutputs.ExecComponentsStep.status
```

LaunchInstance

Diese Schrittaktion startet eine Instance in Ihrem AWS-Konto und wartet, bis der Systems Manager-Agent auf der Instance ausgeführt wird, bevor mit dem nächsten Schritt fortgefahren wird. Die Startaktion verwendet Einstellungen aus Ihren Rezept- und Infrastrukturkonfigurationsressourcen, die Ihrem Image zugeordnet sind. Der zu startende Instance-Typ stammt beispielsweise aus der Infrastrukturkonfiguration. Die Ausgabe ist die Instance-ID der Instance, die sie gestartet hat.

Die `waitFor` Eingabe konfiguriert die Bedingung, die die Anforderung zum Abschluss des Schritts erfüllt.

Standard-Timeout: 60 Minuten

Rollback: Bei Build-Instances führt das Rollback die Aktion aus, die Sie in Ihrer Infrastrukturkonfigurationsressource konfiguriert haben. Standardmäßig werden Build-Instances beendet, wenn die Image-Erstellung fehlschlägt. In der Infrastrukturkonfiguration gibt es jedoch eine Einstellung, um die Build-Instance zur Fehlerbehebung beizubehalten.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
waitFor	Die Bedingung, auf die gewartet werden soll, bevor der Workflow-	String	Ja		Image Builder unterstützt derzeit `ssmAgent`.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
	Schritt abgeschlossen und mit dem nächsten Schritt fortgefahren wird.				

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
instanceId	Die Instance-ID der Instance, die gestartet wurde.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: LaunchStep
  action: LaunchInstance
  onFailure: Abort
  inputs:
    waitFor: ssmAgent
```

Verwenden Sie die Ausgabe der Schrittaktion im Workflow-Dokument.

```
$.stepOutputs.LaunchStep.instanceId
```

RunCommand

Diese Schrittaktion führt ein Befehlsdokument für Ihren Workflow aus. Image Builder verwendet die sendCommand in der Systems Manager API, um sie für Sie auszuführen. Weitere Informationen finden Sie unter AWS Systems Manager Run Command .

Standard-Timeout: 12 Stunden

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der Instance, auf der das Befehlsdokument ausgeführt werden soll.	String	Ja		Dies muss die Ausgabe-Instance-ID aus dem Workflow-Schritt sein, der die Instance für diesen Workflow gestartet hat.
documentName	Der Name des auszuführenden Systems Manager-Befehlsdokuments.	String	Ja		
Parameter	Eine Liste von Schlüssel-Wert-Paaren für alle Parameter, die das Befehlsdo	Wörterbuch<Zeichenfolge, list<Zeichenfolge	Bedingt		

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
	kument benötigt.				
documentVersion	Die auszuführende Version des Befehlsdokuments.	String	Nein	$DEFAULT	

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagerssendCommand, der das Befehlsdokument auf der Instance ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Vom Systems Manager zurückgegebene AusgabesendCommand.	Liste von Zeichenfolgen

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: RunCommandDoc
  action: RunCommand
  onFailure: Abort
  inputs:
    documentName: SampleDocument
```

```
    parameters:
        osPlatform:
          - "linux"
    instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.RunCommandDoc.status
```

RunSysPrep

Diese Schrittaktion verwendet die sendCommand in der Systems Manager-API, um das `AWSEC2-RunSysprep` Dokument für Windows-Instances auszuführen, bevor die Build-Instance für den Snapshot heruntergefahren wird. Diese Aktionen folgen AWS bewährten Methoden zum Verstärken und Bereinigen des Images .

Standard-Timeout: 60 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der Instance, auf der das `AWSEC2-RunSysprep` Dokument ausgeführt werden soll.	String	Ja		Dies muss die Ausgabe-Instance-ID aus dem Workflow-Schritt sein, der die Instance für diesen Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagerssendCommand, der das `AWSEC2-RunSysprep` Dokument auf der Instance ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Vom Systems Manager zurückgegebene AusgabesendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: RunSysprep
  action: RunSysPrep
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.RunSysprep.status
```

SanitizeInstance

Diese Schrittaktion führt das empfohlene Bereinigungsskript für Linux-Instances aus, bevor die Build-Instance für den Snapshot heruntergefahren wird. Das Bereinigungsskript trägt dazu bei, dass das endgültige Image den bewährten Sicherheitsmethoden entspricht und dass Build-Artefakte oder Einstellungen, die nicht auf Ihren Snapshot übertragen werden sollen, entfernt werden. Weitere Informationen zum Skript finden Sie unter Erforderliche Bereinigung nach der Erstellung. Diese Schrittaktion gilt nicht für Container-Images.

Image Builder verwendet die sendCommand in der Systems Manager API, um dieses Skript auszuführen. Weitere Informationen finden Sie unter AWS Systems Manager Run Command .

Standard-Timeout: 60 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der zu bereinigenden Instance.	String	Ja		Dies muss die Ausgabe-Instance-ID aus dem Workflow-Schritt sein, der die Instance für diesen Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagerssendCommand, der das Bereinigungsskript auf der Instance ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String

Ausgabenname	Beschreibung	Typ
output	Vom Systems Manager zurückgegebene AusgabesendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: SanitizeStep
  action: SanitizeInstance
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.SanitizeStep.status
```

TerminateInstance

Diese Schrittaktion beendet die Instance mit der Instance-ID, die als Eingabe übergeben wird.

Standard-Timeout: 30 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der zu beendenden Instance.	String	Ja		

Ausgaben: Für diese Schrittaktion gibt es keine Ausgaben.

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: TerminateInstance
  action: TerminateInstance
  onFailure: Continue
  inputs:
      instanceId.$: i-1234567890abcdef0
```

WaitForAction

Diese Schrittaktion unterbricht den laufenden Workflow und wartet darauf, eine externe Aktion von der Image-BuilderSendWorkflowStepAction-API-Aktion zu erhalten. In diesem Schritt wird ein EventBridge Ereignis in Ihrem Standard- EventBridge Event-Bus mit dem Detailtyp veröffentlicht`EC2 Image Builder Workflow Step Waiting`. Der Schritt kann auch eine SNS-Benachrichtigung senden, wenn Sie einen SNS-Themen-ARN angeben.

Standard-Timeout: 3 Tage

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Eingabename	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
snsTopicArn	Ein optionaler SNS-Themen-ARN, an den eine Benachrichtigung gesendet werden soll, wenn der Workflow-Schritt aussteht.	String	Nein		

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
action	Die Aktion, die die SendWorkflowStepAction API-Aktion zurückgibt.	Zeichenfolge (RESUME oder STOP)
Grund	Der Grund für die zurückgegebene Aktion.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: SendEventAndWait
  action: WaitForAction
  onFailure: Abort
  inputs:
    snsTopicArn: arn:aws:sns:us-west-2:111122223333:ExampleTopic
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.SendEventAndWait.reason
```

Verwenden dynamischer Variablen in Ihrem Workflow-Dokument

Sie können dynamische Variablen in Ihren Workflow-Dokumenten verwenden, um Werte darzustellen, die zur Laufzeit für Ihren Image-Erstellungsprozess variieren. Dynamische Variablenwerte werden als JSONPath-Selektoren mit Strukturknoten dargestellt, die die Zielvariable eindeutig identifizieren.

Struktur der dynamischen JSONPath-Workflow-Variablen

```
$.<document structure>.[<step name>].<variable name>
```

Der erste Knoten nach dem Stamm ($) bezieht sich auf die Struktur des Workflow-`stepOutputs`Dokuments, z. B. oder im Fall von Image-Builder-Systemvariablen,

`imageBuilder`. Die folgende Liste enthält unterstützte JSONPath-Workflow-Dokumentstrukturknoten.

Knoten der Dokumentstruktur

- parameters – Die Workflow-Parameter
- stepOutputs – Ausgaben aus einem Schritt im selben Workflow-Dokument
- workflowOutputs – Ausgaben aus einem Workflow-Dokument, das bereits ausgeführt wurde
- Image Builder – Image-Builder-Systemvariablen

Die `stepOutputs` Dokumentstrukturknoten `parameters` und enthalten einen optionalen Knoten für den Schrittnamen. Dies trägt dazu bei, in allen Schritten eindeutige Variablennamen sicherzustellen.

Der letzte Knoten im JSONPath ist der Name der Zielvariablen, z. B. `instanceId`.

Jeder Schritt kann sich auf die Ausgabe aller vorherigen Schrittaktionen mit diesen dynamischen JSONPath-Variablen beziehen. Dies wird auch als Verketten oder Verweisen auf bezeichnet. Um auf die Ausgabe einer vorherigen Schrittaktion zu verweisen, können Sie die folgende dynamische Variable verwenden.

```
$.stepOutputs.step-name.output-name
```

Beispiel

```
- name: ApplyTestComponents
  action: ExecuteComponents
  onFailure: Abort
  inputs:
    instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"
```

Verwenden von Image-Builder-Systemvariablen

Image Builder bietet die folgenden Systemvariablen, die Sie in Ihrem Workflow-Dokument verwenden können:

Variablenname	Beschreibung	Typ	Beispielwert

Variablenname	Beschreibung	Typ	Beispielwert
cloudWatchLogGruppe	Der Name der CloudWatch Logs-Gruppe für Ausgabeprotokolle. Format: `/aws/imagebuilder/`*`<recipe-name>`*	String	`/aws/imagebuilder/`*`sampleImageRecipe`*
cloudWatchLogStream	Der Name des CloudWatch Logs-Streams für Ausgabeprotokolle.	String	*`1.0.0/1`*
collectImageMetadata	Die Einstellung, die Image Builder anweist, ob Instance-Metadaten erfasst werden sollen.	Boolesch	`true` | `false`
collectImageScanErkenntnisse	Der aktuelle Wert der Einstellung, die es Image Builder ermöglicht, Ergebnisse von Image-Scans zu sammeln.	Boolesch	`true` | `false`
imageBuildNumber	Die Build-Versionsnummer des Images.	Ganzzahl	*`1`*

Variablenname	Beschreibung	Typ	Beispielwert
imageId	Die AMI-ID des Basis-Images.	String	*ami-12345 67890abcdef1*
imageName	Der Name des Image.	String	*sampleImage*
ImageType	Der Image-Ausgabetyp.	String	`AMI` | `Docker`
imageVersionNumber	Die Versionsnummer des Images.	String	*1.0.0*
instanceProfileName	Der Name der Instance-Profilrolle, die Image Builder zum Starten von Build- und Test-Instances verwendet.	String	*SampleImageBuilder InstanceProfileRole*
platform	Die Betriebssystemplattform des erstellten Images.	String	`Linux` | `Windows` | `MacOS`
s3Logs	Ein JSON-Objekt, das die Konfiguration für die S3-Protokolle enthält, die Image Builder schreibt.	JSON-Objekt	{'s3Logs ': {'s3BucketName': '*sample-bucket* ', 's3KeyPrefix': '*ib-logs* '}}'

Variablenname	Beschreibung	Typ	Beispielwert
securityGroups	Die Sicherheitsgruppen-IDs, die für Build- und Test-Instances gelten.	Liste [Zeichenfolge]	*[sg-1234567890abcdef1, sg-11112222333344445]*
sourceImageARN	Der Amazon-Ressourcenname (ARN) der Image Builder-Image-Ressource, die der Workflow für Build- und Testphasen verwendet.	String	arn:aws:imagebuilder:*us-east-1*:*111122223333*:image/*sampleImage*/*1.0.0/1*
subnetId	Die ID des Subnetzes, in dem die Build- und Test-Instances gestartet werden sollen.	String	*subnet-1234567890abcdef1*
terminateInstanceOnFehler	Der aktuelle Wert der Einstellung, die Image Builder anweist, die Instance bei einem Ausfall zu beenden oder zur Fehlerbehebung beizubehalten.	Boolesch	`true` | `false`
workflowPhase	Die aktuelle Phase, die für die Workflow-Ausführung ausgeführt wird.	String	`Build` | `Test`

Variablenname	Beschreibung	Typ	Beispielwert
workingDirectory	Der Pfad zum Arbeitsverzeichnis.	String	`/tmp`

Verwenden von bedingten Anweisungen in Ihren Workflow-Schritten

Bedingte Anweisungen beginnen mit dem Attribut des `if` Anweisungsdokuments. Der endgültige Zweck der `if` Anweisung besteht darin, zu bestimmen, ob die Schrittaktion ausgeführt oder übersprungen werden soll. Wenn die `if` Anweisung zu aufgelöst wird`true`, wird die Schrittaktion ausgeführt. Wenn es zu aufgelöst wird`false`, überspringt Image Builder die Schrittaktion und zeichnet den Schrittstatus `SKIPPED` im Protokoll auf.

Die `if` Anweisung unterstützt Verzweigungsanweisungen (`and`, `or`) und bedingte Modifikatoren (`not`). Es unterstützt auch die folgenden bedingten Anweisungen, die Wertvergleiche (gleich, kleiner als, größer als) basierend auf den Datentypen durchführen, die es vergleicht (Zeichenfolge oder Zahl).

Unterstützte bedingte Anweisungen

- `booleanEquals`
- `numberEquals`
- `numberGreaterThan`
- `numberGreaterThanEquals`
- `numberLessThan`
- `numberLessThanEquals`
- `stringEquals`

Regeln für Verzweigungsanweisungen und bedingte Modifikatoren

Die folgenden Regeln gelten für Verzweigungsanweisungen (`and`, `or`) und bedingte Modifikatoren (`not`).

- Verzweigungsanweisungen und bedingte Modifikatoren müssen in einer Zeile selbst erscheinen.

- Verzweigungsanweisungen und bedingte Modifikatoren müssen den Regeln der Ebene entsprechen.
  - Es kann nur eine Anweisung auf der übergeordneten Ebene geben.
  - Jeder untergeordnete Zweig oder Modifikator startet eine neue Ebene.

  Weitere Informationen zu Ebenen finden Sie unter Verschachtelte Ebenen.
- Jede Verzweigungsanweisung muss mindestens eine untergeordnete bedingte Anweisung, aber nicht mehr als zehn haben.
- Bedingte Modifikatoren arbeiten nur mit einer untergeordneten bedingten Anweisung.

Verschachtelte Ebenen

Bedingte Anweisungen funktionieren auf mehreren Ebenen in einem eigenen Abschnitt. Beispielsweise wird das `if` Anweisungsattribut in Ihrem Workflow-Dokument auf derselben Ebene wie der Schrittname und die Aktion angezeigt. Dies ist die Basis der bedingten Anweisung.

Sie können bis zu vier Ebenen bedingter Anweisungen angeben, aber nur eine Anweisung kann auf der übergeordneten Ebene angezeigt werden. Alle anderen verzweigenden Anweisungen, bedingten Modifikatoren oder bedingten Operatoren werden von dort aus eingerückt, eine Einrückung pro Ebene.

Die folgende Übersicht zeigt die maximale Anzahl verschachtelter Ebenen für eine bedingte Anweisung.

```
base:
  parent:
    - child (level 2)
        - child (level 3)
            child (level 4)
```

`if` Attribut

Das `if` Attribut gibt die bedingte Anweisung als Dokumentattribut an. Dies ist Stufe 0.

Übergeordnete Ebene

Dies ist die erste Verschachtelungsebene für bedingte Anweisungen. Auf dieser Ebene kann es nur eine Anweisung geben. Wenn Sie keine Verzweigungen oder Modifikatoren benötigen, kann dies ein bedingter Operator ohne untergeordnete Anweisungen sein. Diese Stufe verwendet keine Bindestrichnotation, mit Ausnahme bedingter Operatoren.

Untergeordnete Ebenen

Die Stufen zwei bis vier werden als untergeordnete Ebenen betrachtet. Untergeordnete Anweisungen können Verzweigungsanweisungen, bedingte Modifikatoren oder bedingte Operatoren enthalten.

Beispiel: Verschachtelte Ebenen

Das folgende Beispiel zeigt die maximale Anzahl von Ebenen in einer bedingten Anweisung.

```
if:
  and:                            #first level
    - stringEquals: 'my_string'    #second level
      value: 'my_string'
    - and:                        #also second level
        - numberEquals: '1'        #third level
          value: 1
        - not:                     #also third level
            stringEquals: 'second_string'      #fourth level
            value: "diff_string"
```

Verschachtelungsregeln

- Jede Verzweigung oder jeder Modifikator auf untergeordneter Ebene startet eine neue Ebene.
- Jede Ebene ist eingerückt.
- Es kann maximal vier Ebenen geben, darunter eine Anweisung, ein Modifikator oder ein Operator auf der übergeordneten Ebene und bis zu drei zusätzliche Ebenen.

Beispiele

Diese Gruppe von Beispielen zeigt verschiedene Aspekte bedingter Anweisungen.

Verzweigung: und

Die `and` Verzweigungsanweisung arbeitet mit einer Liste von Ausdrücken, die untergeordnete Ausdrücke der Verzweigung sind, die alle mit ausgewertet werden müssen`true`. Image Builder wertet die Ausdrücke in der Reihenfolge aus, in der sie in der Liste erscheinen. Wenn ein Ausdruck mit ausgewertet wird`false`, wird die Verarbeitung beendet und die Verzweigung wird als betrachtet`false`.

Im folgenden Beispiel wird als ausgewertet`true`, da beide Ausdrücke als ausgewertet werden`true`.

```
if:
  and:
    - stringEquals: 'test_string'
      value: 'test_string'
    - numberEquals: 1
      value: 1
```

Verzweigung: oder

Die `or` Verzweigungsanweisung arbeitet mit einer Liste von Ausdrücken, die untergeordnete Ausdrücke des Zweigs sind, von denen mindestens einer mit ausgewertet werden muss`true`. Image Builder wertet die Ausdrücke in der Reihenfolge aus, in der sie in der Liste erscheinen. Wenn ein Ausdruck mit ausgewertet wird`true`, wird die Verarbeitung beendet und die Verzweigung wird als betrachtet`true`.

Das folgende Beispiel ergibt `true`, obwohl der erste Ausdruck ist`false`.

```
if:
  or:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
    - numberEquals: 1
      value: 1
```

Bedingter Modifikator: nicht

Der `not`bedingte Modifikator negiert die bedingten Anweisungen, die untergeordnete Anweisungen des Zweigs sind.

Das folgende Beispiel ergibt `not true`, wenn der Modifikator die `stringEquals` bedingte Anweisung negiert.

```
if:
  not:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
```

Bedingte Anweisung: booleanEquals

Die `booleanEquals`bedingte Anweisung vergleicht boolesche Werte und gibt true zurück, wenn die booleschen Werte genau übereinstimmen.

Das folgende Beispiel bestimmt, ob aktiviert `collectImageScanFindings` ist.

```
if:
  - booleanEquals: true
    value: '$.imagebuilder.collectImageScanFindings'
```

Bedingte Anweisung: stringEquals

Die `stringEquals`bedingte Anweisung vergleicht zwei Zeichenfolgen und gibt true zurück, wenn die Zeichenfolgen genau übereinstimmen. Wenn es sich bei einem der beiden Werte nicht um eine Zeichenfolge handelt, konvertiert Image Builder sie vor dem Vergleich in eine Zeichenfolge.

Im folgenden Beispiel wird die Plattformsystemvariable verglichen, um festzustellen, ob der Workflow auf einer Linux-Plattform ausgeführt wird.

```
if:
  - stringEquals: 'Linux'
    value: '$.imagebuilder.Platform'
```

Bedingte Anweisung: numberEquals

Die `numberEquals`bedingte Anweisung vergleicht zwei Zahlen und gibt true zurück, wenn die Zahlen gleich sind. Die zu vergleichenden Zahlen müssen eines der folgenden Formate haben.

- Ganzzahl
- Gleitkommazahl
- Eine Zeichenfolge, die dem folgenden Regex-Muster entspricht: `^-?[0-9]+(\.)?[0-9]+$`.

Der folgende Beispielvergleichx ergibt `true`.

```
if:
  # Value provider as a number
  numberEquals: 1
  value: '1'

  # Comparison value provided as a string
```

```
  numberEquals: '1'
  value: 1

  # Value provided as a string
  numberEquals: 1
  value: '1'

  # Floats are supported
  numberEquals: 5.0
  value: 5.0

  # Negative values are supported
  numberEquals: -1
  value: -1
```

Importieren und Exportieren von Images virtueller Maschinen (VM) mit EC2 Image Builder

Wenn Sie Ihre VM aus der Virtualisierungsumgebung exportieren, erstellt dieser Prozess einen Satz von einer oder mehreren Festplattencontainerdateien, die als Snapshots der Umgebung, der Einstellungen und der Daten Ihrer VM dienen. Sie können diese Dateien verwenden, um Ihre VM zu importieren, und sie als Basis-Image für Ihre Image-Rezepte verwenden.

Image Builder unterstützt die folgenden Dateiformate für Ihre VM-Festplattencontainer:

- Öffnen des Virtualization Archive (OVA)
- Virtual Machine Disk (VMDK)
- Virtuelle Festplatte (VHD/VHDX)
- Raw

Beim Import werden die Datenträger verwendet, um ein Amazon Machine Image (AMI) und eine Image-Builder-Image-Ressource zu erstellen, von denen beide als Basis-Image für Ihr benutzerdefiniertes Image-Rezept dienen können. Die VM-Datenträger müssen für den Import in S3-Buckets gespeichert werden. Alternativ können Sie aus einem vorhandenen EBS-Snapshot importieren.

In der Image-Builder-Konsole können Sie das Image direkt importieren und dann das Ausgabebild oder AMI in Ihren Rezepten verwenden, oder Sie können Importparameter angeben, wenn Sie Ihr

Rezept oder Ihre Rezeptversion erstellen. Weitere Informationen zum direkten Importieren finden Sie unter Importieren einer VM (Konsole). Weitere Informationen zum Importieren von als Teil Ihres Image-Rezepts finden Sie unter VM-Importkonfiguration.

Importieren einer VM in Image Builder (AWS CLI)

Gehen Sie folgendermaßen vor, um eine VM von Datenträgern in ein AMI zu importieren und eine Image Builder-Image-Ressource zu erstellen, auf die Sie sofort verweisen könnenAWS CLI:

1. Initiieren Sie einen VM-Import mit dem Befehl Amazon EC2 VM Import/Export import-image in der AWS CLI. Notieren Sie sich die Aufgaben-ID, die in der Befehlsantwort zurückgegeben wird. Sie benötigen ihn für den nächsten Schritt. Weitere Informationen finden Sie unter Importieren einer VM als Abbild mit VM Import/Export im Benutzerhandbuch für VM Import/Export.
2. Erstellen einer CLI-Eingabe-JSON-Datei

   Um den Image-Builder-import-vm-imageBefehl zu optimieren, der in der verwendet wirdAWS CLI, erstellen wir eine JSON-Datei, die alle Importkonfigurationen enthält, die wir an den Befehl übergeben möchten.

   > **Note**
   >
   > Die Namenskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Anforderungsparameter der Image-Builder-API-Aktion angegeben ist. Informationen zum Überprüfen der API-Befehlsanforderungsparameter finden Sie unter dem ImportVmImage Befehl in der EC2 Image Builder-API-Referenz .
   > Um die Datenwerte als Befehlszeilenparameter bereitzustellen, beziehen Sie sich auf die Parameternamen, die in der AWS CLI -Befehlsreferenz angegeben sind. auf den Image-Builder-import-vm-imageBefehl als Optionen.

   Im Folgenden finden Sie eine Zusammenfassung der Parameter, die wir in diesem Beispiel angeben:

   - Name (Zeichenfolge, erforderlich) – Der Name für die Image-Builder-Image-Ressource, die als Ausgabe des Imports erstellt werden soll.
   - semanticVersion (Zeichenfolge, erforderlich) – Die semantische Version für das Ausgabebild, die die Version im folgenden Format angibt, mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben: <major>.<minor>.<patch>. Beispiel: `1.0.0` Weitere

Informationen zum semantischen Versioning für Image-Builder-Ressourcen finden Sie unter Semantische Versionsverwaltung.

- description (Zeichenfolge) – Die Beschreibung des Image-Rezepts.
- -Plattform (Zeichenfolge, erforderlich) – Die Betriebssystemplattform für die importierte VM.
- vmImportTaskID (Zeichenfolge, erforderlich) – Die `ImportTaskId` (AWS CLI) aus dem Amazon EC2-VM-Importprozess. Image Builder überwacht den Importvorgang, um das erstellte AMI abzurufen und eine Image-Builder-Image-Ressource zu erstellen, die sofort in Rezepten verwendet werden kann.
- clientToken (Zeichenfolge, erforderlich) – Eine eindeutige Kennung, bei der zwischen Groß- und Kleinschreibung unterschieden wird, die Sie angeben, um die Idempotenz der Anforderung sicherzustellen. Weitere Informationen finden Sie unter Sicherstellen von Idempotenz in der Amazon EC2-API-Referenz.
- Tags (Zeichenfolgezuordnung) – Tags sind Schlüssel-Wert-Paare, die den Importressourcen angefügt sind. Es sind bis zu 50 Schlüssel-Wert-Paare zulässig.

Speichern Sie die Datei als `import-vm-image.json`, um sie im Image-Builder-import-vm-imageBefehl zu verwenden.

```
{
    "name": "example-request",
    "semanticVersion": "1.0.0",
    "description": "vm-import-test",
    "platform": "Linux",
    "vmImportTaskId": "import-ami-01ab234567890cd1e",
    "clientToken": "asz1231231234cs3z",
    "tags": {
     "Usage": "VMIE"
    }
}
```

3. Importieren des Images

   Führen Sie den import-vm-image Befehl mit der Datei aus, die Sie als Eingabe erstellt haben:

```
aws imagebuilder import-vm-image --cli-input-json file://import-vm-image.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Verteilen von VM-Datenträgern aus Ihrem Image-Build (AWS CLI)

Sie können die Verteilung unterstützter VM-Datenträgerformatdateien an S3-Buckets in Zielregionen als Teil Ihres regulären Image-Build-Prozesses einrichten, indem Sie Image-Builder-Verteilungskonfigurationen in der verwendenAWS CLI. Weitere Informationen finden Sie unter Erstellen von Verteilungseinstellungen für VM-Ausgabedatenträger (AWS CLI).

EC2 Image Builder-Ressourcen freigeben

EC2 Image Builder lässt sich in AWS Resource Access Manager (AWS RAM) integrieren, um Ihnen zu ermöglichen, bestimmte Ressourcen mit einem AWS-Konto oder über zu teilenAWS Organizations. EC2 Image Builder-Ressourcen, die freigegeben werden können, sind:

- Komponenten
- Bilder
- Rezepte

Dieser Abschnitt enthält Informationen, die Ihnen helfen, diese EC2 Image Builder-Ressourcen zu teilen.

Abschnittsinhalte

- Freigeben einer Komponente, eines Images oder eines Rezepts
- Aufheben der Freigabe einer freigegebenen Komponente, eines Images oder eines Rezepts
- Identifizieren einer freigegebenen Komponente, eines gemeinsamen Abbilds oder eines gemeinsamen Rezepts
- Berechtigungen für gemeinsam genutzte Komponenten, Abbilder und Rezepte
- Fakturierung und Messung
- Ressourcenlimits

Arbeiten mit freigegebenen Komponenten, Images und Rezepten in EC2 Image Builder

Die gemeinsame Nutzung von Komponenten, Abbildern und Rezepten ermöglicht es Ressourcenbesitzern, Softwarekonfigurationen mit anderen AWS-Konten oder innerhalb einer AWS Organisation gemeinsam zu nutzen. Sie können die Ressourcenfreigabe zentral verwalten und eine Reihe von Konten definieren, für die die Konfiguration freigegeben werden kann.

In diesem Modell gibt das , AWS-Konto das die Komponente, das Bild oder das Rezept besitzt (Eigentümer), sie für andere AWS-Konten (Verbraucher) frei. Konsumenten können ihren Image-Pipelines eine gemeinsam genutzte Komponente zuordnen, um automatisch Updates für die gemeinsam genutzte Komponente, das gemeinsame Image oder das gemeinsame Rezept zu nutzen.

Eine Komponente, ein Image oder ein Rezeptbesitzer kann diese Ressourcen freigeben für:

- Spezifische AWS-Konten innerhalb oder außerhalb seiner Organisation in AWS Organizations.
- Eine Organisationseinheit (OU) innerhalb ihrer Organisation in AWS Organizations.
- Seine gesamte Organisation in AWS Organizations.
- AWS Organizations oder OUs außerhalb seiner Organisation.

Voraussetzungen für die gemeinsame Nutzung von Komponenten, Images und Rezepten

So geben Sie eine Image-Builder-Komponente, ein Image oder ein Rezept frei:

- Sie müssen Eigentümer der Komponente, des Images oder des Rezepts in Ihrem seinAWS-Konto. Sie können keine Ressourcen freigeben, die für Sie freigegeben wurden.

- Der mit verschlüsselten Ressourcen verknüpfte AWS Key Management Service (AWS KMS)-Schlüssel muss explizit mit den Zielkonten, Organisationen oder OUs geteilt werden.
- Um Ihre Image-Builder-Ressourcen mit AWS Organizations und OUs mithilfe von freizugebenAWS RAM, müssen Sie die Freigabe aktivieren. Weitere Informationen finden Sie unter Freigabe für AWS Organizations aktivieren im AWS RAM-Benutzerhandbuch.
- Wenn Sie ein mit verschlüsseltes Image AWS KMS auf Konten in verschiedenen Regionen verteilen, müssen Sie in jeder Zielregion einen KMS-Schlüssel und -Alias erstellen. Darüber hinaus benötigen die Personen, die Instances in diesen Regionen starten werden, Zugriff auf den KMS-Schlüssel, der über die Schlüsselrichtlinie angegeben wird.

Die folgenden Ressourcen, die Image Builder aus Ihrem Pipeline-Build erstellt, gelten nicht als Image-Builder-Ressourcen, sondern sind externe Ressourcen, die Image Builder in Ihrem Konto verteilt, und an die AWS-Regionen, Konten und Organisationen oder Organisationseinheiten (OUs), die Sie in Ihrer Verteilungskonfiguration angeben.

- Amazon Machine Images (AMIs)
- Container-Images, die sich in Amazon ECR befinden

Weitere Informationen zu Verteilungseinstellungen für Ihr AMI finden Sie unter Erstellen und Aktualisieren von AMI-Verteilungskonfigurationen. Weitere Informationen zu Verteilungseinstellungen für Ihr Container-Image in Amazon ECR finden Sie unter Erstellen und Aktualisieren von Verteilungseinstellungen für Container-Images.

Weitere Informationen zur Freigabe Ihres AMI für AWS Organizations und OUs finden Sie unter Freigeben eines AMI für Organisationen oder OUs.

Zugehörige Services

AWS Resource Access Manager

Die Freigabe von Komponenten, Abbildern und Rezepten ist in AWS Resource Access Manager (AWS RAM) integriert. AWS RAM ist ein Service, mit dem Sie Ihre AWS Ressourcen für jedes -AWSKonto oder über freigeben könnenAWS Organizations. Mit geben Sie Ressourcen AWS RAMin Ihrem Besitz frei, indem Sie eine Ressourcenfreigabe erstellen. Eine Ressourcenfreigabe gibt die freizugebenden Ressourcen und die Verbraucher an, für die sie freigegeben werden sollen. Konsumenten können einzelne AWS-Konten, Organisationseinheiten oder eine gesamte Organisation in seinAWS Organizations.

Weitere Informationen zu AWS RAM finden Sie im AWS RAM-Benutzerhandbuch.

Freigeben zwischen Regionen

Gemeinsam genutzte Komponenten, Images und Rezepte können nur in einer bestimmten AWS Region gemeinsam genutzt werden. Wenn Sie diese Ressourcen gemeinsam nutzen, werden sie nicht regionsübergreifend repliziert.

Freigeben einer Komponente, eines Images oder eines Rezepts

Um eine Image-Builder-Komponente, ein Image oder ein Rezept freizugeben, müssen Sie sie einer Ressourcenfreigabe hinzufügen. Eine Ressourcenfreigabe ist eine AWS RAM-Ressource, mit der Sie Ihre Ressourcen in mehreren AWS-Konten gemeinsam nutzen können. Eine Ressourcenfreigabe gibt die freizugebenden Ressourcen und die Konsumenten an, für die sie freigegeben werden. Um die Komponente, das Image oder das Rezept zu einer neuen Ressourcenfreigabe hinzuzufügen, müssen Sie zuerst die Ressourcenfreigabe mithilfe der AWS RAMKonsole erstellen.

Wenn Sie Teil einer Organisation in sind AWS Organizations und die Freigabe innerhalb Ihrer Organisation aktiviert ist, wird Konsumenten in Ihrer Organisation automatisch Zugriff auf die freigegebene Komponente, das Image oder das Rezept gewährt. Andernfalls erhalten Konsumenten eine Einladung zur Teilnahme an der Ressourcenfreigabe und nach Annahme der Einladung wird ihnen Zugriff auf die freigegebene Ressource gewährt.

Die folgenden Optionen sind für die Freigabe Ihrer -Ressourcen verfügbar:

Option 1: Erstellen einer RAM-Ressourcenfreigabe

Wenn Sie eine RAM-Ressourcenfreigabe erstellen, können Sie in einem einzigen Schritt eine Komponente, ein Image oder ein Rezept freigeben, das Sie besitzen. Verwenden Sie eine der folgenden Methoden, um Ihre Ressourcenfreigabe zu erstellen:

- Konsole

  Informationen zum Erstellen Ihrer Ressourcenfreigabe mithilfe der AWS RAM Konsole finden Sie unter Freigeben von AWS Ressourcen, die Sie besitzen im AWS RAM -Benutzerhandbuch.
- AWS CLI

  Um Ihre Ressourcenfreigabe über die AWS RAM Befehlszeilenschnittstelle zu erstellen, führen Sie den Befehl create-resource-share in der ausAWS CLI.

Option 2: Anwenden einer Ressourcenrichtlinie und Hochstufen zu einer RAM-Ressourcenfreigabe

Die zweite Option zum Freigeben Ihrer Ressourcen umfasst zwei Schritte, das Ausführen von Befehlen in der AWS CLI für beide. Im ersten Schritt werden Image-Builder-Befehle in der verwendetAWS CLI, um ressourcenbasierte Richtlinien auf die gemeinsam genutzte Ressource anzuwenden. Der zweite Schritt stuft die Ressource mithilfe des promote-resource-share-created-from-policy AWS RAM Befehls in der zu einer RAM-Ressourcenfreigabe hoch, AWS CLI um sicherzustellen, dass die Ressource für alle Prinzipale sichtbar ist, für die Sie sie freigegeben haben.

1. Anwenden der Ressourcenrichtlinie

   Um die Ressourcenrichtlinie erfolgreich anzuwenden, müssen Sie sicherstellen, dass das Konto, für das Sie die Freigabe durchführen, über die Berechtigung für den Zugriff auf alle zugrunde liegenden Ressourcen verfügt.

   Wählen Sie die Registerkarte aus, die Ihrem Ressourcentyp für den entsprechenden Befehl entspricht.

   Image

   Sie können eine Ressourcenrichtlinie auf ein Image anwenden, damit andere es als Basis-Image in ihren Rezepten verwenden können.

   Führen Sie den put-image-policy Image Builder-Befehl in der ausAWS CLI, um die AWSPrinzipale zu identifizieren, für die das Image freigegeben werden soll.

   ```
   aws imagebuilder put-image-policy --image-arn arn:aws:imagebuilder:us-
   west-2:123456789012:image/my-example-image/2019.12.03/1 --policy
    '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal":
    { "AWS": [ "123456789012" ] }, "Action": ["imagebuilder:GetImage",
    "imagebuilder:ListImages"], "Resource": [ "arn:aws:imagebuilder:us-
   west-2:123456789012:image/my-example-image/2019.12.03/1" ] } ] }'
   ```

   Component

   Sie können eine Ressourcenrichtlinie auf eine Build- oder Testkomponente anwenden, um die kontoübergreifende Freigabe zu ermöglichen. Dieser Befehl erteilt anderen Konten die Berechtigung, Ihre Komponente in ihren Rezepten zu verwenden. Um die Ressourcenrichtlinie erfolgreich anzuwenden, müssen Sie sicherstellen, dass das Konto, mit

dem Sie die Freigabe durchführen, über die Berechtigung für den Zugriff auf alle Ressourcen verfügt, auf die die freigegebene Komponente verweist, z. B. auf Dateien, die auf privaten Repositorys gehostet werden.

Führen Sie den put-component-policy Image Builder-Befehl in der ausAWS CLI, um die AWSPrinzipale zu identifizieren, für die die Komponente freigegeben werden soll.

```
aws imagebuilder put-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/my-example-component/2019.12.03/1 --policy
 '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal":
 { "AWS": [ "123456789012" ] }, "Action": [ "imagebuilder:GetComponent",
 "imagebuilder:ListComponents" ], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-example-component/2019.12.03/1" ] } ] }'
```

Image recipe

Sie können eine Ressourcenrichtlinie auf ein Image-Rezept anwenden, um die kontoübergreifende Freigabe zu ermöglichen. Dieser Befehl erteilt anderen Konten die Berechtigung, Ihr Rezept zum Erstellen von Bildern in ihren Konten zu verwenden. Um die Ressourcenrichtlinie erfolgreich anzuwenden, müssen Sie sicherstellen, dass das Konto, für das Sie die Freigabe durchführen, über die Berechtigung für den Zugriff auf alle Ressourcen verfügt, auf die das Rezept verweist, z. B. das Basis-Image oder ausgewählte Komponenten.

Führen Sie den put-image-recipe-policy Image Builder-Befehl in der ausAWS CLI, um die AWSPrinzipale zu identifizieren, für die das Image freigegeben werden soll.

```
aws imagebuilder put-image-recipe-policy --image-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-
image-recipe/2019.12.03 --policy '{ "Version": "2012-10-17", "Statement":
 [ { "Effect": "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetImageRecipe", "imagebuilder:ListImageRecipes" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-
recipe/2019.12.03" ] } ] }'
```

Container recipe

Sie können eine Ressourcenrichtlinie auf ein Container-Rezept anwenden, um die kontoübergreifende Freigabe zu aktivieren. Dieser Befehl erteilt anderen Konten die Berechtigung, Ihr Rezept zum Erstellen von Bildern in ihren Konten zu verwenden. Um die Ressourcenrichtlinie erfolgreich anzuwenden, müssen Sie sicherstellen, dass das Konto, für

das Sie die Freigabe durchführen, über die Berechtigung für den Zugriff auf alle Ressourcen verfügt, auf die das Rezept verweist, z. B. das Basis-Image oder ausgewählte Komponenten.

Führen Sie den put-container-recipe-policy Image Builder-Befehl in der ausAWS CLI, um die AWSPrinzipale zu identifizieren, für die das Image freigegeben werden soll.

```
aws imagebuilder put-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-example-
container-recipe/2021.12.03 --policy '{ "Version": "2012-10-17", "Statement":
 [ { "Effect": "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetContainerRecipe", "imagebuilder:ListContainerRecipes" ],
 "Resource": [ "arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-
example-container-recipe/2021.12.03" ] } ] }'
```

> **Note**
>
> Um die richtigen Richtlinien für die Freigabe und Aufhebung der Freigabe einer Ressource festzulegen, muss der Ressourcenbesitzer über `-imagebuilder:put*`Berechtigungen verfügen.

2. Hochstufen als RAM-Ressourcenfreigabe

   Um sicherzustellen, dass die Ressource für alle Prinzipale sichtbar ist, für die Sie sie freigegeben haben, führen Sie den promote-resource-share-created-from-policy AWS RAM Befehl in der ausAWS CLI.

Aufheben der Freigabe einer freigegebenen Komponente, eines Images oder eines Rezepts

Um die Freigabe einer freigegebenen Komponente, eines Images oder eines Rezepts, das Sie besitzen, aufzuheben, müssen Sie sie aus der Ressourcenfreigabe entfernen. Hierfür können Sie die AWS Resource Access Manager-Konsole oder die AWS CLI verwenden.

> **Note**
>
> Um die Freigabe einer Komponente, eines Images oder eines Rezepts aufzuheben, kann der Konsument keine Abhängigkeiten von ihm haben. Der Konsument muss alle Abhängigkeiten von den gemeinsam genutzten Ressourcen entfernen, bevor der Besitzer sie aufheben kann.

So heben Sie die Freigabe einer freigegebenen Komponente, eines Images oder eines Rezepts in Ihrem Besitz mithilfe der AWS Resource Access Manager Konsole auf

Siehe Aktualisieren einer Ressourcenfreigabe im AWS RAM-Benutzerhandbuch.

So heben Sie die Freigabe einer freigegebenen Komponente, eines Images oder eines Rezepts in Ihrem Besitz mithilfe der auf AWS CLI

Verwenden Sie den disassociate-resource-share Befehl , um die Freigabe der Ressource zu beenden.

Identifizieren einer freigegebenen Komponente, eines gemeinsamen Abbilds oder eines gemeinsamen Rezepts

Besitzer und Verbraucher können freigegebene Komponenten, Bilder und Image-Rezepte mithilfe von Image-Builder-Befehlen in der identifizierenAWS CLI.

Identifizieren einer freigegebenen Komponente

Führen Sie den Befehl list- components aus, um eine Liste der Komponenten abzurufen, die Sie besitzen, und der Komponenten, die für Sie freigegeben werden. Der Befehl get- component zeigt die AWS-Konto ID des Eigentümers der Komponente an.

Identifizieren eines freigegebenen Abbilds

Führen Sie den Befehl list-images aus, um eine Liste der Images abzurufen, die Sie besitzen, und Images, die für Sie freigegeben werden. Der Befehl get-image zeigt die AWS-Konto ID des Image-Besitzers an.

Identifizieren eines freigegebenen Container-Images

Führen Sie den Befehl list-images aus, um eine Liste der Images abzurufen, die Sie besitzen, und Images, die für Sie freigegeben werden. Der Befehl get-image zeigt die AWS-Konto ID des Image-Besitzers an.

Identifizieren eines freigegebenen Image-Rezepts

Führen Sie den list-image-recipes Befehl aus, um eine Liste der Image-Rezepte, die Sie besitzen, und der Image-Rezepte abzurufen, die für Sie freigegeben werden. Der get-image-recipe Befehl zeigt die AWS-Konto ID des Besitzers des Image-Rezepts an.

Identifizieren eines freigegebenen Container-Rezepts

Führen Sie den list-container-recipes Befehl aus, um eine Liste der Container-Rezepte, die Sie besitzen, und der Container-Rezepte abzurufen, die für Sie freigegeben sind. Der get-container-recipe Befehl zeigt die AWS-Konto ID des Besitzers des Container-Rezepts an.

Berechtigungen für gemeinsam genutzte Komponenten, Abbilder und Rezepte

Berechtigungen für Besitzer

Besitzer können eine freigegebene Komponente, ein Image oder ein Image-Rezept erst löschen, wenn sie nicht mehr freigegeben ist. Ein Besitzer kann die Freigabe dieser Ressourcen erst aufheben, wenn keiner der Konsumenten von ihnen abhängig ist.

Berechtigungen für Konsumenten

Konsumenten können eine Komponente, ein Image oder ein Image-Rezept lesen, aber sie in keiner Weise ändern. Sie können diese Ressourcen nicht anzeigen oder ändern, wenn sie anderen Konsumenten oder dem Eigentümer der Ressource gehören. Konsumenten können gemeinsam genutzte Komponenten und Bilder in Image-Rezepten verwenden, um benutzerdefinierte Bilder zu erstellen. Konsumenten können freigegebene Image-Rezepte verwenden, um ihre eigenen benutzerdefinierten Images zu erstellen.

Fakturierung und Messung

Für die Nutzung von EC2 Image Builder fallen keine Gebühren an.

Ressourcenlimits

Gemeinsam genutzte Komponenten, Images und Image-Rezepte werden nur auf die entsprechenden Ressourcenlimits des Besitzers angerechnet. Die Ressourcenlimits der Konsumenten werden von den Ressourcen, die für sie freigegeben wurden, nicht beeinflusst.

EC2 Image Builder-Ressourcen markieren

Das Markieren Ihrer Ressourcen kann nützlich sein, um Ressourcenkosten oder andere Kategorien zu filtern und zu verfolgen. Sie können den Zugriff auch anhand von Tags steuern. Weitere Informationen zur Tag-basierten Autorisierung finden Sie unter Autorisierung basierend auf Image-Builder-Tags

Image Builder unterstützt die folgenden dynamischen Tags:

- `- {{imagebuilder:buildDate}}`

  Löst sich zum Erstellungsdatum/-zeitpunkt zur Erstellungszeit auf.

- `- {{imagebuilder:buildVersion}}`

  Löst sich in eine Build-Version auf. Dabei handelt es sich um eine Zahl am Ende eines Image-Builder-ARN. zeigt beispielsweise `"arn:aws:imagebuilder:us-west-2:123456789012:component/myexample-component/2019.12.02/1"` die Build-Version als an`1`.

Inhalt

Markieren einer Ressource (AWS CLI)

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um eine Ressource in EC2 Image Builder hinzuzufügen und zu markieren. Sie müssen die Tags `resourceArn` und angeben, die darauf angewendet werden sollen.

Der `tag-resource.json` Beispielinhalt lautet wie folgt:

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-example-pipeline",
    "tags": {
        "KeyName": "KeyValue"
```

```
    }
}
```

Führen Sie den folgenden Befehl aus, der auf die vorhergehende `tag-resource.json` Datei verweist.

```
aws imagebuilder tag-resource --cli-input-json file://tag-resource.json
```

Aufheben der Markierung einer Ressource (AWS CLI)

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um ein Tag aus einer Ressource zu entfernen. Sie müssen die Schlüssel `resourceArn` und angeben, um das Tag zu entfernen.

Der `untag-resource.json` Beispielinhalt lautet wie folgt:

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-
example-pipeline",
    "tagKeys": [
        "KeyName"
    ]
}
```

Führen Sie den folgenden Befehl aus, der auf die vorhergehende `untag-resource.json` Datei verweist.

```
aws imagebuilder untag-resource --cli-input-json file://untag-resource.json
```

Auflisten aller Tags für eine bestimmte Ressource (AWS CLI)

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um alle Tags für eine bestimmte Ressource aufzulisten.

```
aws imagebuilder list-tags-for-resource --resource-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

Löschen von EC2 Image Builder-Ressourcen

Ihre Image-Builder-Umgebung benötigt wie Ihr Heim regelmäßige Wartungsarbeiten, um Ihnen zu helfen, das zu finden, was Sie benötigen, und Ihre Aufgaben zu erledigen, ohne dass Sie durch ein Überladen laufen müssen. Bereinigen Sie regelmäßig temporäre Ressourcen, die Sie zum Testen erstellt haben. Andernfalls vergessen Sie möglicherweise diese Ressourcen und erinnern sich später nicht daran, wofür sie verwendet wurden. Dann ist es möglicherweise nicht klar, ob Sie sie sicher loslegen können.

Durch das Löschen von Ressourcen werden keine Amazon EC2-AMIs oder Amazon-ECR-Container-Images gelöscht, die während des Image-Build-Prozesses erstellt werden. Sie müssen diese separat bereinigen, indem Sie die entsprechenden Amazon EC2- oder Amazon ECR-Konsolenaktionen oder API- oder -AWS CLIBefehle verwenden.

> **Tip**
>
> Um Abhängigkeitsfehler beim Löschen von Ressourcen zu vermeiden, stellen Sie sicher, dass Sie Ihre Ressourcen in der folgenden Reihenfolge löschen:
>
> 1. Image-Pipeline
> 2. Image-Rezept
> 3. Alle verbleibenden Ressourcen

Löschen von Ressourcen mithilfe der -AWSManagementkonsole

Gehen Sie folgendermaßen vor, um eine Image-Pipeline und ihre Ressourcen zu löschen:

Löschen der Pipeline

1. Um eine Liste der Build-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.
2. Aktivieren Sie das Kontrollkästchen neben Pipeline-Name, um die Pipeline auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Pipelines im Menü Aktionen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen des Rezepts

1. Um eine Liste der unter Ihrem Konto erstellten Rezepte anzuzeigen, wählen Sie im Navigationsbereich Image-Rezepte aus.
2. Aktivieren Sie das Kontrollkästchen neben Rezeptname, um das Rezept auszuwählen, das Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Rezepte im Menü Aktionen die Option Rezept löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Infrastrukturkonfiguration löschen

1. Um eine Liste der Infrastrukturkonfigurationen anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Infrastrukturkonfiguration auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Infrastrukturkonfigurationen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen von Verteilungseinstellungen

1. Um eine Liste der unter Ihrem Konto erstellten Verteilungseinstellungen anzuzeigen, wählen Sie im Navigationsbereich Verteilungseinstellungen aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Verteilungseinstellungen auszuwählen, die Sie für dieses Tutorial erstellt haben.
3. Wählen Sie oben im Bereich Verteilungseinstellungen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Ein Image löschen

1. Um eine Liste der unter Ihrem Konto erstellten Images anzuzeigen, wählen Sie im Navigationsbereich Images aus.

2. Wählen Sie die Image-Version für das Image aus, das Sie entfernen möchten. Dadurch wird die Seite Image-Build-Versionen geöffnet.
3. Aktivieren Sie das Kontrollkästchen neben Version für jedes Image, das Sie löschen möchten. Sie können mehr als eine Image-Version gleichzeitig auswählen.
4. Wählen Sie oben im Bereich Image-Build-Versionen die Option Version löschen aus.
5. Um den Löschvorgang zu bestätigen, geben Sie `Delete` in das Feld ein und wählen Sie Löschen aus.

Löschen einer Image-Pipeline mithilfe der AWS CLI

Die folgenden Beispiele zeigen, wie Sie Image-Builder-Ressourcen mithilfe der löschenAWS CLI. Wie bereits erwähnt, müssen Ressourcen in der folgenden Reihenfolge gelöscht werden, um Abhängigkeitsfehler zu vermeiden:

1. Image-Pipeline
2. Image-Rezept
3. Alle verbleibenden Ressourcen

Image-Pipeline löschen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie eine Image-Pipeline löschen, indem Sie ihren ARN angeben.

```
aws imagebuilder delete-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

Image-Rezept löschen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie ein Image-Rezept löschen, indem Sie seinen ARN angeben.

```
aws imagebuilder delete-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2019.12.03
```

Löschen einer Infrastrukturkonfiguration

Das folgende Beispiel zeigt, wie Sie eine Infrastrukturkonfigurationsressource löschen, indem Sie ihren ARN angeben.

```
aws imagebuilder delete-infrastructure-configuration --infrastructure-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-
infrastructure-configuration
```

Löschen von Verteilungseinstellungen

Das folgende Beispiel zeigt, wie Sie eine Verteilungseinstellungen-Ressource löschen, indem Sie ihren ARN angeben.

```
aws imagebuilder delete-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

Ein Image löschen

Das folgende Beispiel zeigt, wie Sie eine Image-Build-Version löschen, indem Sie ihren ARN angeben.

```
aws imagebuilder delete-image --image-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.02/1
```

Löschen einer Komponente

Das folgende Beispiel zeigt, wie Sie einen imagebuilder -CLI-Befehl verwenden, um eine Komponenten-Build-Version zu löschen, indem Sie ihren ARN angeben.

```
aws imagebuilder delete-component --component-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-
component/2019.12.02/1
```

> **Important**
>
> Stellen Sie sicher, dass es keine Rezepte gibt, die auf die Komponenten-Build-Version verweisen, bevor Sie sie löschen. Andernfalls kann es zu Pipeline-Fehlern kommen.

Verwalten von EC2 Image Builder-Pipelines mit der Konsole

Image-Builder-Image-Pipelines bieten ein Automatisierungs-Framework zum Erstellen und Verwalten benutzerdefinierter AMIs und Container-Images. Pipelines bieten die folgenden Funktionen:

- Zusammenstellen des Basis-Images, der Komponenten für Erstellung und Tests, der Infrastrukturkonfiguration und der Verteilungseinstellungen.
- Vereinfachen Sie die Planung für automatisierte Wartungsprozesse mithilfe der `Schedule builder` im Konsolenassistenten oder geben Sie Cron-Ausdrücke für wiederkehrende Updates Ihrer Images ein.
- Aktivieren Sie die Änderungserkennung für das Basis-Image und die Komponenten, um geplante Builds automatisch zu überspringen, wenn keine Änderungen vorgenommen werden.
- Aktivieren Sie die regelbasierte Automatisierung über Amazon EventBridge.

> **Note**
>
> Weitere Informationen zur Verwendung der EventBridge -API zum Anzeigen oder Ändern von Regeln finden Sie in der Amazon EventBridge -API-Referenz. Weitere Informationen zur Verwendung EventBridge von events Befehlen in der AWS CLI zum Anzeigen oder Ändern von Regeln finden Sie unter -Ereignisse in der AWS CLI -Befehlsreferenz.

Inhalt

Auflisten und Anzeigen von Pipeline-Details

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder-Image-Pipelines anzeigen können.

Pipeline-Details

- Auflisten von Image-Pipelines (AWS CLI)
- Abrufen von Image-Pipeline-Details (AWS CLI)

Auflisten von Image-Pipelines (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den list-image-pipelines Befehl in verwenden, AWS CLI um alle Ihre Image-Pipelines aufzulisten.

```
aws imagebuilder list-image-pipelines
```

Abrufen von Image-Pipeline-Details (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den get-image-pipeline Befehl in der verwenden, AWS CLI um die Details zu einer Image-Pipeline über ihren ARN abzurufen.

```
aws imagebuilder get-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

Erstellen und Aktualisieren von AMI-Image-Pipelines

Sie können AMI-Image-Pipelines über die Image-Builder-Konsole, über die Image-Builder-API oder mit -imagebuilderBefehlen in der einrichten, konfigurieren und verwaltenAWS CLI. Sie können den Assistenten zum Erstellen einer Image-Pipeline-Konsole verwenden, um Sie durch die folgenden Schritte zu führen:

- Geben Sie Pipeline-Details wie Name, Beschreibung und Ressourcen-Tags an.
- Wählen Sie ein AMI-Image-Rezept aus, das ein Basis-Image aus schnellstartverwalteten Images oder Images enthält, die Sie erstellt haben oder die für Sie freigegeben wurden. Das Rezept enthält auch Komponenten, die die folgenden Aufgaben auf den EC2-Instances ausführen, die Image Builder zum Erstellen Ihres Images verwendet:

  - Hinzufügen und Entfernen von Software
  - Anpassen von Einstellungen und Skripts
  - Ausgewählte Tests ausführen
- Geben Sie Workflows an, um Image-Build- und Testschritte zu konfigurieren, die Ihre Pipeline ausführt.
- Definieren Sie die Infrastrukturkonfiguration für Ihre Pipeline mit Standardeinstellungen oder Einstellungen, die Sie selbst konfigurieren. Die Konfiguration umfasst den Instance-Typ und das Schlüsselpaar, die für Ihr Image verwendet werden sollen, Sicherheits- und Netzwerkeinstellungen, Einstellungen für Protokollspeicher und Fehlerbehebung sowie SNS-Benachrichtigungen.

  Dies ist ein optional erSchritt. Image Builder verwendet Standardeinstellungen für Ihre Infrastrukturkonfiguration, wenn Sie die Konfiguration nicht selbst definieren.
- Definieren Sie Verteilungseinstellungen, um Ihre Images an AWS Zielregionen und -konten zu übermitteln. Sie können einen KMS-Schlüssel für die Verschlüsselung angeben, die AMI-Freigabe oder die Lizenzkonfiguration konfigurieren oder eine Startvorlage für die von Ihnen verteilten AMIs konfigurieren.

  Dies ist ein optional erSchritt. Wenn Sie die Konfiguration nicht selbst definieren, verwendet Image Builder die Standardbenennung für Ihr Ausgabe-AMI und verteilt das AMI an die Quellregion. Die Quellregion ist die Region, in der Sie die Pipeline ausführen.

Weitere Informationen und ein step-by-step Tutorial zur Verwendung des Assistenten zum Erstellen einer Image-Pipeline-Konsole mit Standardwerten, sofern angegeben, finden Sie unter Erstellen einer Image-Pipeline mit dem EC2 Image Builder-Konsolenassistenten.

Inhalt

Erstellen einer AMI-Image-Pipeline (AWS CLI)

Sie können eine AMI-Image-Pipeline mit einer JSON-Datei erstellen, die Konfigurationsdetails als Eingabe für den create-image-pipeline Befehl in der enthältAWS CLI.

Wie oft Ihre Pipeline ein neues Image erstellt, um ausstehende Updates aus Ihrem Basis-Image und Ihren Komponenten zu integrieren`schedule`, hängt von der von Ihnen konfigurierten ab. Ein `schedule` hat die folgenden Attribute:

- `scheduleExpression` – Legt den Zeitplan fest, wann Ihre Pipeline ausgeführt wird, um die zu bewerten `pipelineExecutionStartCondition` und zu bestimmen, ob sie einen Build starten soll. Der Zeitplan ist mit Cron-Ausdrücken konfiguriert. Weitere Informationen zum Formatieren eines Cron-Ausdrucks in Image Builder finden Sie unter Verwenden von Cron-Ausdrücken in EC2 Image Builder.
- `pipelineExecutionStartCondition` – Bestimmt, ob Ihre Pipeline den Build starten soll. Gültige Werte sind:
  - `EXPRESSION_MATCH_ONLY` – Ihre Pipeline erstellt jedes Mal ein neues Image, wenn der Cron-Ausdruck mit der aktuellen Zeit übereinstimmt.
  - `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE` – Ihre Pipeline startet keinen neuen Image-Build, es sei denn, es gibt ausstehende Änderungen an Ihrem Basis-Image oder Ihren Komponenten.

Wenn Sie den create-image-pipeline Befehl in der ausführenAWS CLI, sind viele der Konfigurationsressourcen optional. Einige der Ressourcen haben jedoch bedingte Anforderungen, je nachdem, welche Art von Image die Pipeline erstellt. Die folgenden Ressourcen sind für AMI-Image-Pipelines erforderlich:

- Image-Rezept-ARN
- ARN der Infrastrukturkonfiguration

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-image-pipeline.json`:

   ```
   {
    "name": "MyWindows2019Pipeline",
    "description": "Builds Windows 2019 Images",
    "enhancedImageMetadataEnabled": true,
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
   ```

```
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-image-
pipeline.json
```

Aktualisieren von AMI-Image-Pipelines (Konsole)

Nachdem Sie eine Image-Builder-Image-Pipeline für Ihr AMI-Image erstellt haben, können Sie über die Image-Builder-Konsole Änderungen an den Infrastrukturkonfigurations- und Verteilungseinstellungen vornehmen.

Um eine Image-Pipeline mit einem neuen Image-Rezept zu aktualisieren, müssen Sie die verwendenAWS CLI. Weitere Informationen finden Sie unter Aktualisieren von AMI-Image-Pipelines (AWS CLI) in diesem Handbuch.

Wählen Sie eine vorhandene Image-Builder-Pipeline

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der unter Ihrem Konto erstellten Image-Pipelines anzuzeigen, wählen Sie im Navigationsbereich Image-Pipelines aus.

   > **Note**
   >
   > Die Liste der Image-Pipelines enthält einen Indikator für den Typ des Ausgabe-Images, das von der Pipeline erstellt wird – AMI oder Docker.

3. Um Details anzuzeigen oder eine Pipeline zu bearbeiten, wählen Sie den Link Pipeline-Name. Dadurch wird die Detailansicht für die Pipeline geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Pipeline-Namen aktivieren und dann Details anzeigen auswählen.

Pipeline-Details

Die Seite mit den Pipeline-Details enthält die folgenden Abschnitte:

Übersicht

Im Abschnitt oben auf der Seite werden die wichtigsten Details für die Pipeline zusammengefasst, die bei geöffneter Detailregisterkarte sichtbar sind. Die in diesem Abschnitt angezeigten Details können nur auf den jeweiligen Detailregisterkarten bearbeitet werden.

Detail-Registerkarten

- Ausgabebilder – Zeigt Ausgabebilder an, die die Pipeline erstellt hat.
- Image recipe – Zeigt Rezeptdetails an. Nachdem Sie ein Rezept erstellt haben, können Sie es nicht mehr bearbeiten. Sie müssen eine neue Version des Rezepts auf der Seite Image-Rezepte in der

Image-Builder-Konsole oder mithilfe von Image-Builder-Befehlen in der erstellenAWS CLI. Weitere Informationen finden Sie unter Rezepte verwalten.

- Infrastrukturkonfiguration – Zeigt bearbeitbare Informationen zur Konfiguration Ihrer Build-Pipeline-Infrastruktur an.
- Verteilungseinstellungen – Zeigt bearbeitbare Informationen für die AMI-Verteilung an.
- -EventBridge Regeln – zeigt für den ausgewählten Event Bus EventBridge Regeln an, die auf die aktuelle Pipeline abzielen. Enthält Ereignisbus erstellen und Regelaktionen erstellen, die mit der EventBridge Konsole verknüpft sind. Weitere Informationen zu dieser Registerkarte finden Sie unter Verwenden von EventBridge Regeln.

Bearbeiten der Infrastrukturkonfiguration für Ihre Pipeline

Die Infrastrukturkonfiguration enthält die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Infrastrukturkonfiguration.
- Die IAM-Rolle, die dem Instance-Profil zugeordnet werden soll.
- AWS Infrastruktur , einschließlich des Instance-Typs und eines SNS-Themas für Benachrichtigungen.
- VPC, Subnetz und Sicherheitsgruppen .
- Fehlerbehebungseinstellungen, einschließlich Beenden der Instance bei Fehlern, das Schlüsselpaar für die Verbindung und ein optionaler S3-Bucket-Speicherort für Instance-Protokolle.

Gehen Sie folgendermaßen vor, um die Infrastrukturkonfiguration auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie die Registerkarte Infrastrukturkonfiguration aus.
2. Wählen Sie oben rechts im Bereich Konfigurationsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, Aktualisierungen zu speichern, die Sie an Ihrer Infrastrukturkonfiguration vorgenommen haben, wählen Sie Änderungen speichern aus.

Bearbeiten der Verteilungseinstellungen für Ihre Pipeline

Die Verteilungseinstellungen enthalten die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Verteilungskonfiguration.
- Regionseinstellungen für die Regionen, in denen Sie Ihr Image verteilen. Region 1 ist standardmäßig die Region, in der Sie die Pipeline erstellt haben. Mit der Schaltfläche Region hinzufügen können Sie Regionen für die Verteilung hinzufügen und alle Regionen außer Region 1 entfernen.

  Zu den Regionseinstellungen gehören:

  - Zielregion
  - Der Name des Ausgabe-AMI
  - Startberechtigungen für -, - und -Konten, mit denen sie geteilt werden können
  - Zugeordnete Lizenzen (Zuordnen von Lizenzkonfigurationen)

  > **Note**
  >
  > License Manager-Einstellungen werden nicht über AWS Regionen hinweg repliziert, die in Ihrem Konto aktiviert werden müssen, z. B. zwischen den Regionen `ap-east-1` (Hongkong) und `me-south-1` (Bahrain).

Gehen Sie folgendermaßen vor, um Ihre Verteilungseinstellungen auf der Pipeline-Detailseite zu bearbeiten:

1. Wählen Sie die Registerkarte Verteilungseinstellungen aus.
2. Wählen Sie oben rechts im Bereich Verteilungsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, Ihre Aktualisierungen zu speichern, wählen Sie Änderungen speichern aus.

Bearbeiten des Build-Zeitplans für Ihre Pipeline

Die Seite Pipeline bearbeiten enthält die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Pipeline.
- Verbesserte Metadatensammlung . Diese Option ist standardmäßig aktiviert. Um es auszuschalten, deaktivieren Sie das Kontrollkästchen Erweiterte Metadatensammlung aktivieren.

- Der Build-Zeitplan für Ihre Pipeline. Sie können Ihre Zeitplanoptionen und alle Einstellungen hier ändern.

Gehen Sie folgendermaßen vor, um Ihre Pipeline auf der Pipeline-Detailseite zu bearbeiten:

1. Wählen Sie oben rechts auf der Seite mit den Pipeline-Details Aktionen und dann Pipeline bearbeiten aus.
2. Wenn Sie bereit sind, Ihre Aktualisierungen zu speichern, wählen Sie Änderungen speichern aus.

> **Note**
>
> Weitere Informationen zum Planen Ihres Builds mit Cron-Ausdrücken finden Sie unter Verwenden von Cron-Ausdrücken in EC2 Image Builder.

Aktualisieren von AMI-Image-Pipelines (AWS CLI)

Sie können eine AMI-Image-Pipeline mithilfe einer JSON-Datei als Eingabe für den update-image-pipeline Befehl in der aktualisierenAWS CLI. Um die JSON-Datei zu konfigurieren, benötigen Sie Amazon-Ressourcennamen (ARNs), um auf die folgenden vorhandenen Ressourcen zu verweisen:

- Image-Pipeline zum Aktualisieren
- Image-Rezept
- Infrastrukturkonfiguration
- Distribution Settings (Einstellungen für die Verteilung)

Sie können eine AMI-Image-Pipeline mit dem update-image-pipeline Befehl in der AWS CLI wie folgt aktualisieren:

> **Note**
>
> UpdateImagePipeline unterstützt keine selektiven Updates für die Pipeline. Sie müssen alle erforderlichen Eigenschaften in der Aktualisierungsanforderung angeben, nicht nur die Eigenschaften, die sich geändert haben.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Name`create-component.json`:

```
 {
 "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
pipeline/my-example-pipeline",
 "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
example-recipe/2019.12.08",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 120
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "DISABLED"
}
```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-pipeline.json
```

Erstellen und Aktualisieren von Container-Image-Pipelines

Sie können Container-Image-Pipelines über die Image-Builder-Konsole, über die Image-Builder-API oder mit imagebuilder Befehlen in der einrichten, konfigurieren und verwaltenAWS CLI. Der Assistent zum Erstellen einer Image-Pipeline-Konsole bietet Startartefakte und führt Sie durch die folgenden Schritte:

- Auswählen eines Basis-Images aus schnellstartfähigen verwalteten Images, Amazon ECR- oder Docker Hub-Repositorys
- Hinzufügen und Entfernen von Software
- Anpassen von Einstellungen und Skripts
- Ausgewählte Tests ausführen
- Erstellen Sie ein Dockerfile mit vorkonfigurierten Build-Zeitvariablen.
- Verteilen von Images an -AWSRegionen

Weitere Informationen und ein step-by-step Tutorial zur Verwendung des Assistenten zum Erstellen einer Image-Pipeline-Konsole finden Sie unter Erstellen einer Container-Image-Pipeline mit dem EC2 Image Builder-Konsolenassistenten.

Inhalt

Erstellen einer Container-Image-Pipeline (AWS CLI)

Sie können eine Container-Image-Pipeline mit einer JSON-Datei als Eingabe für den create-image-pipeline Befehl in der erstellenAWS CLI.

Wie oft Ihre Pipeline ein neues Image erstellt, um ausstehende Updates aus Ihrem Basis-Image und Ihren Komponenten zu integrieren`schedule`, hängt von der von Ihnen konfigurierten ab. Ein `schedule` hat die folgenden Attribute:

- `scheduleExpression` – Legt den Zeitplan fest, wann Ihre Pipeline ausgeführt wird, um die zu bewerten `pipelineExecutionStartCondition` und zu bestimmen, ob sie einen Build starten soll. Der Zeitplan ist mit Cron-Ausdrücken konfiguriert. Weitere Informationen zum Formatieren eines Cron-Ausdrucks in Image Builder finden Sie unter Verwenden von Cron-Ausdrücken in EC2 Image Builder.
- `pipelineExecutionStartCondition` – Bestimmt, ob Ihre Pipeline den Build starten soll. Gültige Werte sind:
  - `EXPRESSION_MATCH_ONLY` – Ihre Pipeline erstellt jedes Mal ein neues Image, wenn der Cron-Ausdruck mit der aktuellen Zeit übereinstimmt.
  - `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE` – Ihre Pipeline startet keinen neuen Image-Build, es sei denn, es gibt ausstehende Änderungen an Ihrem Basis-Image oder Ihren Komponenten.

Wenn Sie den create-image-pipeline Befehl in der ausführenAWS CLI, sind viele der Konfigurationsressourcen optional. Einige der Ressourcen haben jedoch bedingte Anforderungen, je nachdem, welche Art von Image die Pipeline erstellt. Die folgenden Ressourcen sind für Container-Image-Pipelines erforderlich:

- Container-Rezept-ARN
- ARN der Infrastrukturkonfiguration

Wenn Sie beim Ausführen des create-image-pipeline Befehls keine Verteilungskonfigurationsressource angeben, wird das Ausgabe-Image in dem ECR-Repository gespeichert, das Sie als Ziel-Repository in Ihrem Container-Rezept in der Region angeben, in der Sie den Befehl ausführen. Wenn Sie eine Verteilungskonfigurationsressource für Ihre Pipeline einschließen, wird das Ziel-Repository verwendet, das Sie für die erste Region in der Verteilung angegeben haben.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-image-pipeline.json`:

```
{
 "name": "MyWindows2019Pipeline",
 "description": "Builds Windows 2019 Images",
 "enhancedImageMetadataEnabled": true,
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-example-recipe/2020.12.03",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-image-
pipeline.json
```

Aktualisieren einer Container-Image-Pipeline (Konsole)

Nachdem Sie eine Image-Builder-Container-Image-Pipeline für Ihr Docker-Image erstellt haben, können Sie über die Image-Builder-Konsole Änderungen an den Infrastrukturkonfigurations- und Verteilungseinstellungen vornehmen.

Um eine Container-Image-Pipeline mit einem neuen Container-Rezept zu aktualisieren, müssen Sie die verwendenAWS CLI. Weitere Informationen finden Sie unter Container-Image-Pipelines aktualisieren (AWS CLI) in diesem Handbuch.

Wählen Sie eine vorhandene Image-Builder-Docker-Image-Pipeline

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der unter Ihrem Konto erstellten Image-Pipelines anzuzeigen, wählen Sie im Navigationsbereich Image-Pipelines aus.

   > **Note**
   >
   > Die Liste der Image-Pipelines enthält einen Indikator für den Typ des Ausgabe-Images, das von der Pipeline erstellt wird – AMI oder Docker.

3. Um Details anzuzeigen oder eine Pipeline zu bearbeiten, wählen Sie den Link Pipeline-Name. Dadurch wird die Detailansicht für die Pipeline geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Pipeline-Namen aktivieren und dann Details anzeigen auswählen.

Pipeline-Details

Die Seite mit den Details zur EC2 Image Builder-Pipeline enthält die folgenden Abschnitte:

Übersicht

Im Abschnitt oben auf der Seite werden die wichtigsten Details für die Pipeline zusammengefasst, die bei geöffneter Detailregisterkarte sichtbar sind. Die in diesem Abschnitt angezeigten Details können nur auf den jeweiligen Detailregisterkarten bearbeitet werden.

Detail-Registerkarten

- Ausgabebilder – Zeigt Ausgabebilder an, die die Pipeline erstellt hat.
- Container-Rezept – Zeigt Rezeptdetails an. Nachdem Sie ein Rezept erstellt haben, können Sie es nicht mehr bearbeiten. Sie müssen eine neue Version des Rezepts auf der Seite Container-Rezepte erstellen. Weitere Informationen finden Sie unter Erstellen einer neuen Version eines Container-Rezepts.
- Infrastrukturkonfiguration – Zeigt bearbeitbare Informationen zur Konfiguration Ihrer Build-Pipeline-Infrastruktur an.
- Verteilungseinstellungen – Zeigt bearbeitbare Informationen für die Docker-Image-Verteilung an.
- -EventBridge Regeln – zeigt für den ausgewählten Event Bus EventBridge Regeln an, die auf die aktuelle Pipeline abzielen. Enthält Ereignisbus erstellen und Regelaktionen erstellen, die mit der EventBridge Konsole verknüpft sind. Weitere Informationen zu dieser Registerkarte finden Sie unter Verwenden von EventBridge Regeln.

Bearbeiten der Infrastrukturkonfiguration für Ihre Pipeline

Die Infrastrukturkonfiguration enthält die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Infrastrukturkonfiguration.
- Die IAM-Rolle, die dem Instance-Profil zugeordnet werden soll.
- AWS Infrastruktur , einschließlich des Instance-Typs und eines SNS-Themas für Benachrichtigungen.
- VPC, Subnetz und Sicherheitsgruppen .
- Fehlerbehebungseinstellungen, einschließlich Beenden der Instance bei Fehlern, das Schlüsselpaar für die Verbindung und ein optionaler S3-Bucket-Speicherort für Instance-Protokolle.

Gehen Sie folgendermaßen vor, um die Infrastrukturkonfiguration auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie die Registerkarte Infrastrukturkonfiguration aus.

2. Wählen Sie oben rechts im Bereich Konfigurationsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, Aktualisierungen zu speichern, die Sie an Ihrer Infrastrukturkonfiguration vorgenommen haben, wählen Sie Änderungen speichern aus.

Bearbeiten der Verteilungseinstellungen für Ihre Pipeline

Die Verteilungseinstellungen enthalten die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Verteilungseinstellungen.
- Regionseinstellungen für die Regionen, in denen Sie Ihr Image verteilen. Region 1 ist standardmäßig die Region, in der Sie die Pipeline erstellt haben. Mit der Schaltfläche Region hinzufügen können Sie Regionen für die Verteilung hinzufügen und alle Regionen außer Region 1 entfernen.

  Zu den Regionseinstellungen gehören:

  - Zielregion
  - Der Service ist standardmäßig auf „ECR“ eingestellt und kann nicht bearbeitet werden.
  - Repository-Name – der Name Ihres Ziel-Repositorys (ohne den Amazon-ECR-Speicherort). Der Repository-Name mit dem Speicherort würde beispielsweise wie folgt aussehen:

    ```
    <account-id>.dkr.ecr.<region>.amazonaws.com/<repository-name>
    ```

    > **Note**
    >
    > Wenn Sie den Repository-Namen ändern, werden nur die Images, die nach der Namensänderung erstellt wurden, unter dem neuen Namen hinzugefügt. Alle vorherigen Images, die Ihre Pipeline erstellt hat, verbleiben im ursprünglichen Repository.

Gehen Sie folgendermaßen vor, um Ihre Verteilungseinstellungen auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie die Registerkarte Verteilungseinstellungen aus.
2. Wählen Sie oben rechts im Bereich Verteilungsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, Aktualisierungen zu speichern, die Sie an Ihren Verteilungseinstellungen vorgenommen haben, wählen Sie Änderungen speichern aus.

Bearbeiten des Build-Zeitplans für Ihre Pipeline

Die Seite Pipeline bearbeiten enthält die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Pipeline.
- Verbesserte Metadatensammlung . Diese Option ist standardmäßig aktiviert. Um es auszuschalten, deaktivieren Sie das Kontrollkästchen Erweiterte Metadatensammlung aktivieren.
- Der Build-Zeitplan für Ihre Pipeline. Sie können Ihre Zeitplanoptionen und alle Einstellungen in diesem Abschnitt ändern.

Gehen Sie folgendermaßen vor, um Ihre Pipeline auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie oben rechts auf der Seite mit den Pipeline-Details Aktionen und dann Pipeline bearbeiten aus.
2. Wenn Sie bereit sind, Ihre Aktualisierungen zu speichern, wählen Sie Änderungen speichern aus.

> **Note**
>
> Weitere Informationen zum Planen Ihres Builds mit Cron-Ausdrücken finden Sie unter Verwenden von Cron-Ausdrücken in EC2 Image Builder.

Container-Image-Pipelines aktualisieren (AWS CLI)

Sie können eine Container-Image-Pipeline mithilfe einer JSON-Datei als Eingabe für den update-image-pipeline Befehl in aktualisierenAWS CLI. Um die JSON-Datei zu konfigurieren, benötigen Sie Amazon-Ressourcennamen (ARNs), um auf die folgenden vorhandenen Ressourcen zu verweisen:

- Image-Pipeline zum Aktualisieren
- Container-Rezept
- Infrastrukturkonfiguration
- Verteilungseinstellungen (falls in der aktuellen Pipeline enthalten)

> **Note**
>
> Wenn die Ressource für Verteilungseinstellungen enthalten ist, hat das ECR-Repository, das als Ziel-Repository in den Verteilungseinstellungen für die Region angegeben ist, in der der Befehl ausgeführt wird (Region 1), Vorrang vor dem Ziel-Repository, das im Container-Rezept angegeben ist.

Gehen Sie wie folgt vor, um eine Container-Image-Pipeline mit dem update-image-pipeline Befehl in der zu aktualisierenAWS CLI:

> **Note**
>
> UpdateImagePipeline unterstützt keine selektiven Updates für die Pipeline. Sie müssen alle erforderlichen Eigenschaften in der Aktualisierungsanforderung angeben, nicht nur die Eigenschaften, die sich geändert haben.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-component.json`:

   ```
   {
    "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
   pipeline/my-example-pipeline",
    "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
   recipe/my-example-recipe/2020.12.08",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
   west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
   configuration",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
   west-2:123456789012:distribution-configuration/my-example-distribution-
   configuration",
    "imageTestsConfiguration": {
     "imageTestsEnabled": true,
     "timeoutMinutes": 120
    },
    "schedule": {
   ```

```
  "scheduleExpression": "cron(0 0 * * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "DISABLED"
}
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-pipeline.json
```

Konfigurieren von Image-Workflows für Ihre EC2 Image Builder-Pipeline

Mit Image-Workflows können Sie die Workflows anpassen, die Ihre Pipeline ausführt, um Images an Ihre Bedürfnisse anzupassen und zu testen. Die Workflows, die Sie definieren, werden im Kontext des Image-Builder-Workflow-Frameworks ausgeführt. Weitere Informationen zu den Phasen, aus denen das Workflow-Framework besteht, finden Sie unter Verwalten von Build- und Test-Workflows für EC2-Image-Builder-Images.

Workflow erstellen

Build-Workflows werden während der `Build` Phase des Workflow-Frameworks ausgeführt. Sie können nur einen Build-Workflow für Ihre Pipeline angeben. Oder Sie können den Build vollständig überspringen, um eine reine Testpipeline zu konfigurieren.

Test-Workflow

Testworkflows werden während der `Test` Phase des Workflow-Frameworks ausgeführt. Sie können bis zu zehn Testworkflows für Ihre Pipeline angeben. Sie können Tests auch vollständig überspringen, wenn Sie nur möchten, dass Ihre Pipeline erstellt wird.

Definieren von Testgruppen für Testworkflows

Testworkflows werden innerhalb von Testgruppen definiert. Sie können bis zu zehn Testworkflows für Ihre Pipeline ausführen. Sie entscheiden, ob Sie die Test-Workflows in einer bestimmten Reihenfolge oder so viele wie möglich gleichzeitig ausführen möchten. Wie sie ausgeführt werden, hängt davon ab, wie Sie Ihre Testgruppen definieren. Die folgenden Szenarien zeigen verschiedene Möglichkeiten, wie Sie Ihre Testworkflows definieren können.

> **Note**
>
> Wenn Sie die Konsole verwenden, um Workflows zu erstellen, empfehlen wir Ihnen, sich Zeit zu nehmen, um zu planen, wie Sie Ihre Testworkflows ausführen möchten, bevor Sie Ihre Testgruppen definieren. In der Konsole können Sie Testworkflows und -gruppen hinzufügen oder entfernen, aber Sie können sie nicht neu anordnen.

Szenario 1: Führen Sie jeweils einen Testworkflow aus

Um alle Ihre Testworkflows einzeln auszuführen, können Sie bis zu zehn Testgruppen konfigurieren, jede mit einem einzigen Testworkflow. Testgruppen werden nacheinander in der Reihenfolge ausgeführt, in der Sie sie Ihrer Pipeline hinzufügen. Auf diese Weise können Sie sicherstellen, dass Ihre Test-Workflows einzeln in einer bestimmten Reihenfolge ausgeführt werden.

Szenario 2: Ausführen mehrerer Testworkflows gleichzeitig

Wenn die Reihenfolge unerheblich ist und Sie so viele Testworkflows wie möglich gleichzeitig ausführen möchten, können Sie eine einzelne Testgruppe konfigurieren und die maximale Anzahl von Testworkflows einfügen. Image Builder startet bis zu fünf Test-Workflows gleichzeitig und startet zusätzliche Test-Workflows, sobald andere abgeschlossen sind. Wenn es Ihr Ziel ist, Ihre Test-Workflows so schnell wie möglich auszuführen, ist dies eine Möglichkeit, dies zu tun.

Szenario 3: Mischen und Abgleichen

Wenn Sie ein gemischtes Szenario mit einigen Testworkflows haben, die gleichzeitig ausgeführt werden können, und einige, die einzeln ausgeführt werden sollten, können Sie Ihre Testgruppen so konfigurieren, dass dieses Ziel erreicht wird. Das einzige Limit für die Konfiguration Ihrer Testgruppen ist die maximale Anzahl von Testworkflows, die für Ihre Pipeline ausgeführt werden können.

Workflow-Parameter in einer Image-Builder-Pipeline festlegen (Konsole)

Workflow-Parameter funktionieren für Build-Workflows und Test-Workflows auf die gleiche Weise. Wenn Sie eine Pipeline erstellen oder aktualisieren, wählen Sie Build- und Test-Workflows aus, die Sie einbeziehen möchten. Wenn Sie Parameter im Workflow-Dokument für einen von Ihnen ausgewählten Workflow definiert haben, zeigt Image Builder diese im Bereich Parameter an. Der Bereich ist für Workflows ausgeblendet, für die keine Parameter definiert sind.

Jeder Parameter zeigt die folgenden Attribute an, die Ihr Workflow-Dokument definiert hat:

- Name (nicht bearbeitbar) – Der Name des Parameters.
- Typ (nicht bearbeitbar) – Der Datentyp für den Parameterwert.
- Wert – Der Wert für den Parameter. Sie können den Parameterwert bearbeiten, um ihn für Ihre Pipeline festzulegen.

Geben Sie die IAM-Servicerolle an, die Image Builder zum Ausführen von Workflow-Aktionen verwendet

Zugriff auf Services

Um Image-Workflows auszuführen, benötigt Image Builder die Berechtigung zum Ausführen von Workflow-Aktionen. Sie können die AWSServiceRoleForImageBuilder serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle für den Servicezugriff wie folgt angeben.

- Konsole – Wählen Sie im Pipeline-Assistenten Schritt 3 Image-Erstellung definieren die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle aus der Liste der IAM-Rollen im Bereich Servicezugriff aus.
- Image-Builder-API – Geben Sie in der CreateImage Aktionsanforderung die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle als Wert für den `executionRole` Parameter an.

Weitere Informationen zum Erstellen einer Servicerolle finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen -AWSService im AWS Identity and Access Management - Benutzerhandbuch.

Ausführen Ihrer Image-Pipeline

Wenn Sie die Option für den manuellen Zeitplan für Ihre Pipeline ausgewählt haben, wird sie nur ausgeführt, wenn Sie den Build manuell starten. Wenn Sie eine der Optionen für die automatische Planung ausgewählt haben, können Sie sie auch manuell ausführen, zwischen regelmäßig geplanten Durchläufen. Wenn Sie beispielsweise eine Pipeline haben, die normalerweise einmal im Monat ausgeführt wird, aber zwei Wochen nach der vorherigen Ausführung eine Aktualisierung einer Ihrer Komponenten integrieren müssen, können Sie Ihre Pipeline manuell ausführen.

Console

Um Ihre Pipeline auf der Pipeline-Detailseite in der Image-Builder-Konsole auszuführen, wählen Sie im Menü Aktionen oben auf der Seite die Option Pipeline ausführen aus. Oben auf der Seite wird eine Statusmeldung angezeigt, die Sie darüber informiert, dass Ihre Pipeline gestartet wurde oder wenn ein Fehler auftritt.

1. Wählen Sie oben links auf der Seite mit den Pipeline-Details im Menü Aktionen die Option Pipeline ausführen aus.
2. Sie können den aktuellen Status Ihrer Pipeline auf der Registerkarte Ausgabebilder in der Spalte Status sehen.

AWS CLI

Das folgende Beispiel zeigt, wie Sie den start-image-pipeline-execution Befehl in verwenden, AWS CLI um eine Image-Pipeline manuell zu starten. Wenn Sie diesen Befehl ausführen, erstellt die Pipeline ein neues Image und verteilt es.

```
aws imagebuilder start-image-pipeline-execution --image-pipeline-arn
 arn:aws:imagebuilder:us-west-2:111122223333:image-pipeline/my-example-pipeline
```

Informationen dazu, welche Ressourcen erstellt werden, wenn die Build-Pipeline ausgeführt wird, finden Sie unter Erstellte Ressourcen.

Verwenden von Cron-Ausdrücken in EC2 Image Builder

Verwenden Sie Cron-Ausdrücke für EC2 Image Builder, um ein Zeitfenster für die Aktualisierung Ihres Images mit Updates einzurichten, die für das Basis-Image und die Komponenten Ihrer Pipeline gelten. Das Zeitfenster für Ihre Pipeline-Aktualisierung beginnt mit der Zeit, die Sie im Cron-Ausdruck festgelegt haben. Sie können die Zeit in Ihrem Cron-Ausdruck auf die Minute festlegen. Ihr Pipeline-Build kann am oder nach der Startzeit ausgeführt werden.

Es kann manchmal einige Sekunden oder bis zu einer Minute dauern, bis Ihr Build gestartet wird.

> **Note**
>
> Cron-Ausdrücke verwenden UTC (Universal Coordinated Time). Weitere Informationen zur UTC-Zeit und zum Ermitteln des Offsets für Ihre Zeitzone finden Sie unter Zeitzonenabkürzungen – Weltweite Liste.

Unterstützte Werte für Cron-Ausdrücke in Image Builder

EC2 Image Builder verwendet ein Cron-Format, das aus sechs Pflichtfeldern besteht. Jede ist durch ein Leerzeichen dazwischen von den anderen getrennt, ohne Leerzeichen am Anfang oder am Ende:

<Minute> <Hour> <Day> <Month> <Day of the week> <Year>

Die folgende Tabelle zeigt die Werte, die für erforderliche Cron-Einträge unterstützt werden.

Unterstützte Werte für Cron-Ausdrücke

Feld	Werte	Platzhalter
Minute	`0-59`	`, - * /`
Stunde	`0-23`	`, - * /`
Tag	`1-31`	`, - * ? / L W`
Monat	`1-12` oder `jan-dec`	`, - * /`
Wochentag	`1-7` oder `sun-sat`	`, - * ? L #`
Jahr	`1970-2199`	`, - * /`

Platzhalter

In der folgenden Tabelle wird beschrieben, wie Image Builder Platzhalter für Cron-Ausdrücke verwendet. Beachten Sie, dass es nach der von Ihnen angegebenen Zeit bis zu einer Minute dauern kann, bis der Build beginnt.

Unterstützte Platzhalter für Cron-Ausdrücke

Platzhalter	Beschreibung
,	Das Platzhalterzeichen , (Komma) schließt zusätzliche Werte ein. Im Feld Monat `jan,feb,mar` enthält Januar, Februar und März.
-	Das Platzhalterzeichen - (Bindestrich) gibt einen Bereich an. schließt im Feld Tag des Monats die `1-15` Tage 1 bis 15 des angegeben en Monats ein.
*	Der Platzhalter * (Sternchen) enthält alle gültigen Werte für das Feld.
?	Der Platzhalter ? (Fragezeichen) gibt an, dass der Feldwert von einer anderen Einstellung abhängt. Wenn in den ay-of-week Feldern Tag und D einer angegeben ist oder alle möglichen Werte enthält (`*`), muss der andere ein sein?. Sie können nicht beides angeben. Wenn Sie beispielsweise einen 7 in das Feld Tag eingeben (den Build am siebten Tag des Monats ausführen), muss die ay-of-week Position D einen enthalten?.
/	Das Platzhalterzeichen / (Schrägstrich) steht für schrittweise Steigerungen. Wenn Ihr Build beispielsweise jeden zweiten Tag ausgeführ t werden soll, geben Sie `*/2` in das Feld Tag ein.

Platzhalter	Beschreibung
L	Der L-Platzhalter in einem der Tagesfelder gibt den letzten Tag an: 28–31 für den Tag des Monats, je nachdem, was der Monat ist, oder Sonntag für den Wochentag.
W	Der W-Platzhalter im Feld Day-of-month gibt einen Wochentag an. Wenn Sie im ay-of-month Feld D eine Zahl vor der eingeben`W`, bedeutet dies, dass Sie den Wochentag anvisieren möchten, der diesem Tag am nächsten ist. Wenn Sie beispielsweise angeben`3W`, soll Ihr Build an dem Wochentag ausgeführt werden, der dem dritten Tag des Monats am nächsten ist.
#	Das # (Hash) ist nur für das Wochentagfeld zulässig und muss mit einer Zahl zwischen 1 und 5 übereinstimmen. Die Zahl gibt an, welche Wochen in einem bestimmten Monat für die Ausführung des Builds gelten. Wenn Ihr Build beispielsweise am zweiten Freitag jedes Monats ausgeführt werden soll, verwenden Sie `fri#2` für das Feld Wochentag.

Einschränkungen

- Sie können die ay-of-week Felder D ay-of-month und D nicht im selben Cron-Ausdruck angeben. Wenn Sie einen Wert oder * in einem dieser Felder angeben, müssen Sie einen ? im anderen verwenden.
- cron-Ausdrücke werden mit einer Ausführungsrate ab einer Minute unterstützt, kürzere Intervalle sind nicht möglich.

Beispiele für Cron-Ausdrücke in EC2 Image Builder

Cron-Ausdrücke werden für die Image-Builder-Konsole anders eingegeben als für die API oder CLI. Um Beispiele anzuzeigen, wählen Sie die Registerkarte aus, die für Sie gilt.

Image Builder console

Die folgenden Beispiele zeigen Cron-Ausdrücke, die Sie in die Konsole für Ihren Build-Zeitplan eingeben können. Die UTC-Zeit wird mit einer 24-Stunden-Uhr angegeben.

Täglich um 10:00 Uhr (UTC) ausführen

```
0 10 * * ? *
```

Täglich um 12:15 Uhr (UTC) ausführen

```
15 12 * * ? *
```

Täglich um Mitternacht (UTC) ausführen

```
0 0 * * ? *
```

Ausführung jeden Wochentagmorgen um 10:00 Uhr (UTC)

```
0 10 ? * 2-6 *
```

Ausführung jeden Wochentag um 18 Uhr (UTC)

```
0 18 ? * mon-fri *
```

Ausführung um 8:00 Uhr (UTC) am ersten Tag eines jeden Monats

```
0 8 1 * ? *
```

Ausführung am zweiten Dienstag jedes Monats um 22:30 Uhr (UTC)

```
30 22 ? * tue#2 *
```

> **Tip**
>
> Wenn Sie nicht möchten, dass Ihr Pipeline-Auftrag bis zum nächsten Tag erweitert wird, während er ausgeführt wird, stellen Sie sicher, dass Sie die Zeit für Ihren Build berücksichtigen, wenn Sie die Startzeit angeben.

API/CLI

Die folgenden Beispiele zeigen Cron-Ausdrücke, die Sie mithilfe von CLI-Befehlen oder API-Anforderungen für Ihren Build-Zeitplan eingeben können. Es wird nur der Cron-Ausdruck angezeigt.

Täglich um 10:00 Uhr (UTC) ausführen

```
cron(0 10 * * ? *)
```

Täglich um 12:15 Uhr (UTC) ausführen

```
cron(15 12 * * ? *)
```

Täglich um Mitternacht (UTC) ausführen

```
cron(0 0 * * ? *)
```

Ausführung jeden Wochentagmorgen um 10:00 Uhr (UTC)

```
cron(0 10 ? * 2-6 *)
```

Ausführung jeden Wochentag um 18:00 Uhr (UTC)

```
cron(0 18 ? * mon-fri *)
```

Ausführung um 8:00 Uhr (UTC) am ersten Tag eines jeden Monats

```
cron(0 8 1 * ? *)
```

Ausführung am zweiten Dienstag jedes Monats um 22:30 Uhr (UTC)

```
cron(30 22 ? * tue#2 *)
```

> **Tip**
>
> Wenn Sie nicht möchten, dass Ihr Pipeline-Auftrag bis zum nächsten Tag erweitert wird, während er ausgeführt wird, stellen Sie sicher, dass Sie die Zeit für Ihren Build berücksichtigen, wenn Sie die Startzeit angeben.

Rate-Ausdrücke in EC2 Image Builder

Ein Rate-Ausdruck beginnt, wenn Sie eine Regel für ein geplantes Ereignis erstellen und mit dem definierten Zeitplan ausführen.

Rate-Ausdrücke bestehen aus zwei Pflichtfeldern. Die Felder werden durch ein Leerzeichen voneinander getrennt.

Syntax

```
rate(value unit)
```

Wert

Eine positive Zahl.

Einheit

Die Zeiteinheit. Für Werte von 1 werden verschiedene Einheiten benötigt, z. B. `minute`, ebenso für Werte über 1, z. B. `minutes`.

Zulässige Werte: Minute | Minuten | Stunde | Stunden | Tag | Tage

Einschränkungen

Wenn der Wert gleich 1 ist, muss die Einheit im Singular stehen. Wenn die Werte größer als 1 sind, muss die Einheit im Plural stehen. Beispielsweise sind `rate(1 hours)` und `rate(5 hour)` ungültige, `rate(1 hour)` und `rate(5 hours)` jedoch gültige Werte.

Verwenden von EventBridge Regeln mit Image-Builder-Pipelines

Ereignisse aus einer Vielzahl von - AWS und -Partnerservices werden nahezu in Echtzeit an Amazon EventBridge Event Buses gestreamt. Sie können auch benutzerdefinierte Ereignisse generieren und Ereignisse aus Ihren eigenen Anwendungen an senden EventBridge. Die Event Buses verwenden Regeln, um zu bestimmen, wohin Ereignisdaten weitergeleitet werden sollen.

Image-Builder-Pipelines sind als EventBridge Regelziele verfügbar, was bedeutet, dass Sie eine Image-Builder-Pipeline basierend auf Regeln ausführen können, die Sie erstellen, um auf Ereignisse im Bus oder nach einem Zeitplan zu reagieren.

> **Note**
>
> Event Buses sind spezifisch für eine Region. Die Regel und das Ziel müssen sich in derselben Region befinden.

Inhalt

EventBridge Begriffe

Dieser Abschnitt enthält eine Zusammenfassung der Begriffe, die Ihnen helfen, zu verstehen, wie in Ihre Image-Builder-Pipelines EventBridge integriert wird.

Ereignis

Beschreibt eine Änderung in einer Umgebung, die sich auf eine oder mehrere Anwendungsressourcen auswirken kann. Die Umgebung kann eine AWS Umgebung, ein SaaS-Partnerservice oder eine SaaS-Partneranwendung oder eine Ihrer Anwendungen oder Services sein. Sie können geplante Ereignisse auch nach einem Zeitplan einrichten.

Event Bus

Eine Pipeline, die Ereignisdaten von Anwendungen und Services empfängt.

Quelle

Der Service oder die Anwendung, die das Ereignis an den Event Bus gesendet hat.

Ziel

Eine Ressource oder ein Endpunkt, die bzw. der EventBridge aufgerufen wird, wenn er mit einer Regel übereinstimmt, wobei Daten aus dem Ereignis an das Ziel übermittelt werden.

Regel

Eine Regel ordnet eintreffende Ereignisse zu und leitet diese zur Verarbeitung an Ziele weiter. Eine einzelne Regel kann ein Ereignis an mehrere Ziele senden, die dann parallel ausgeführt werden können. Regeln basieren entweder auf einem Ereignismuster oder einem Zeitplan.

Muster

Ein Ereignismuster definiert die Ereignisstruktur und die Felder, mit denen eine Regel übereinstimmt, um die Zielaktion zu initiieren.

Plan

Zeitplanregeln führen eine Aktion nach einem Zeitplan aus, z. B. die Ausführung einer Image-Builder-Pipeline, um ein Image vierteljährlich zu aktualisieren. Es gibt zwei Arten von Zeitplanausdrücken:

- Cron-Ausdrücke – Ordnen Sie bestimmte Planungskriterien mithilfe der Cron-Syntax zu, die einfache Kriterien beschreiben kann, z. B. wöchentlich an einem bestimmten Tag. Sie können auch komplexere Kriterien festlegen, z. B. vierteljährlich am fünften Tag des Monats zwischen 2 und 4 Uhr.
- Rate-Ausdrücke – Geben Sie ein reguläres Intervall an, in dem das Ziel aufgerufen wird, z. B. alle 12 Stunden.

Anzeigen von EventBridge Regeln für Ihre Image-Builder-Pipeline

Auf der Registerkarte EventBridge Regeln auf der Detailseite der Image-Builder-Image-Pipelines werden EventBridge Event Buses angezeigt, auf die Ihr Konto Zugriff hat, sowie die Regeln für den ausgewählten Event Bus, die für die aktuelle Pipeline gelten. Diese Registerkarte verweist auch direkt auf die EventBridge Konsole zum Erstellen neuer Ressourcen.

Aktionen, die mit der EventBridge Konsole verknüpft sind

- Erstellen eines Event Bus
- Regel erstellen

Weitere Informationen zu EventBridgefinden Sie in den folgenden Themen im Amazon- EventBridge Benutzerhandbuch.

- Was ist Amazon? EventBridge
- Amazon EventBridge -Event-Buses
- Amazon- EventBridge Ereignisse
- Amazon- EventBridge Regeln

Verwenden von EventBridge Regeln zum Planen eines Pipeline-Builds

In diesem Beispiel erstellen wir eine neue Zeitplanregel für den Standard-Event Bus unter Verwendung eines Ratenausdrucks. Die Regel in diesem Beispiel generiert alle 90 Tage ein Ereignis im Event Bus. Das Ereignis initiiert einen Pipeline-Build, um das Image zu aktualisieren.

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der unter Ihrem Konto erstellten Image-Pipelines anzuzeigen, wählen Sie im Navigationsbereich Image-Pipelines aus.

   > **Note**
   >
   > Die Liste der Image-Pipelines enthält einen Indikator für den Typ des Ausgabe-Images, das von der Pipeline erstellt wird – AMI oder Docker.

3. Um Details anzuzeigen oder eine Pipeline zu bearbeiten, wählen Sie den Link Pipeline-Name. Dadurch wird die Detailansicht für die Pipeline geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Pipeline-Namen aktivieren und dann Details anzeigen auswählen.

4. Öffnen Sie die Registerkarte EventBridge Regeln.
5. Behalten Sie den Standard-Event-Bus bei, der im Bereich Event Bus vorausgewählt ist.
6. Wählen Sie Regel erstellen aus. Dadurch gelangen Sie zur Seite Regel erstellen in der Amazon-EventBridge Konsole.
7. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Der Regelname muss innerhalb des Event Bus für die ausgewählte Region eindeutig sein.
8. Wählen Sie im Bereich Muster definieren die Option Zeitplan aus. Dadurch wird das Panel erweitert, wobei die Option Feste Rate für jede Option ausgewählt ist.
9. Geben Sie `90` in das erste Feld ein und wählen Sie Tage aus der Dropdown-Liste aus.
10. Führen Sie die folgenden Aktionen im Bereich Ziele auswählen aus:

    a. Wählen Sie `EC2 Image Builder` aus der Dropdown-Liste Ziel aus.

    b. Um die Regel auf eine Image-Builder-Pipeline anzuwenden, wählen Sie die Ziel-Pipeline aus der Dropdown-Liste Image Pipeline aus.

c. EventBridge benötigt die Berechtigung, einen Build für die ausgewählte Pipeline zu initiieren. Behalten Sie in diesem Beispiel die Standardoption für Erstellen einer neuen Rolle für diese spezifische Ressource bei.

d. Wählen Sie Add target (Ziel hinzufügen) aus.

11. Wählen Sie Create (Erstellen) aus.

> **Note**
>
> Weitere Informationen zu den Einstellungen für Regeln für Ratenausdrücke, die in diesem Beispiel nicht behandelt werden, finden Sie unter Rate-Ausdrücke im Amazon- EventBridge Benutzerhandbuch.

Integrieren von Produkten und Services in EC2 Image Builder

EC2 Image Builder lässt sich in AWS Marketplace und andere AWS-Services - und -Anwendungen integrieren, um Ihnen zu helfen, robuste, sichere benutzerdefinierte Maschinenabbilder zu erstellen.

Produkte

Image-Builder-Rezepte können Image-Produkte von AWS Marketplace und von Image Builder verwaltete Komponenten integrieren, um wie folgt spezielle Build- und Testfunktionen bereitzustellen.

- AWS Marketplace Image-Produkte – Verwenden Sie ein Image-Produkt von AWS Marketplace als Basis-Image in Ihrem Rezept, um die organisatorischen Standards zu erfüllen, z. B. CIS Hardening. Wenn Sie ein Rezept über die Image-Builder-Konsole erstellen, können Sie aus Ihren vorhandenen Abonnements auswählen oder nach einem bestimmten Produkt in suchenAWS Marketplace. Wenn Sie ein Rezept aus der Image-Builder-API, -CLI oder dem -SDK erstellen, können Sie einen Amazon-Ressourcennamen (ARN) für das Image-Produkt angeben, der als Basis-Image verwendet werden soll.
- AWSTOE components – Komponenten, die Sie in Ihren Rezepten angeben, können Build- und Testaktionen durchführen, z. B. um Software zu installieren oder eine Compliance-Validierung durchzuführen. Einige Image-Produkte, von denen Sie abonnieren, enthalten AWS Marketplace möglicherweise eine unterstützende Komponente, die Sie in Ihren Rezepten verwenden können. Die CIS-gehärteten Images enthalten eine passende AWSTOE Komponente, die Sie in Ihrem Rezept verwenden können, um CIS-Benchmarks-Level-1-Richtlinien für Ihre Konfiguration durchzusetzen.

> **Note**
>
> Weitere Informationen zu Compliance-bezogenen Produkten finden Sie unter Compliance-Produkte für Ihre Image-Builder-Images.

Services

Image Builder lässt sich in die folgenden integrieren, AWS-Services um detaillierte Ereignismetriken, Protokollierung und Überwachung bereitzustellen. Diese Informationen helfen Ihnen dabei, Ihre

Aktivitäten zu verfolgen, Probleme mit der Image-Erstellung zu beheben und Automatisierungen basierend auf Ereignisbenachrichtigungen zu erstellen.

- AWS CloudTrail – Überwachen Sie Image-Builder-Ereignisse, die an gesendet werden CloudTrail. Weitere Informationen zu CloudTrailfinden Sie unter Was ist AWS CloudTrail? im AWS CloudTrail - Benutzerhandbuch.
- Amazon CloudWatch Logs – Überwachen, Speichern und Zugriff auf Ihre Image-Builder-Protokolldateien. Optional können Sie Ihre Protokolle in einem S3-Bucket speichern. Weitere Informationen zu - CloudWatch Protokollen finden Sie unter Was ist Amazon CloudWatch Logs? im Amazon- CloudWatch Logs-Benutzerhandbuch.
- Amazon EventBridge – Stellen Sie eine Verbindung zu einem Stream von Echtzeit-Ereignisdaten aus Image-Builder-Aktivitäten in Ihrem Konto her. Weitere Informationen zu EventBridgefinden Sie unter Was ist Amazon EventBridge? im Amazon- EventBridge Benutzerhandbuch.
- Amazon Inspector – Ermitteln Sie Schwachstellen in Ihren Software- und Netzwerkeinstellungen mit automatischen Scans für die EC2-Test-Instance, die Image Builder startet, um ein neues Image zu erstellen. Image Builder speichert Ergebnisse für Ihre Ausgabe-Image-Ressource, sodass Sie nach dem Beenden Ihrer Test-Instance untersuchen und korrigieren können. Weitere Informationen zu Scans und Preisen finden Sie unter Was ist Amazon Inspector ?im Amazon Inspector-Benutzerhandbuch.

  Amazon Inspector kann Ihre ECR-Repositorys auch scannen, wenn Sie das erweiterte Scannen konfigurieren. Weitere Informationen finden Sie unter Scannen von Amazon-ECR-Container-Images im Amazon Inspector-Benutzerhandbuch.

  > **Note**
  >
  > Amazon Inspector ist ein kostenpflichtiges Feature.

- AWS Marketplace – Sehen Sie sich eine Liste Ihrer aktuellen AWS Marketplace Produktabonnements an und suchen Sie direkt in Image Builder nach Image-Produkten. Sie können auch ein Image-Produkt verwenden, das Sie als Basis-Image für ein Image-Builder-Rezept abonniert haben. Weitere Informationen zum Verwalten von AWS Marketplace Abonnements finden Sie im AWS Marketplace -Käuferhandbuch.
- Amazon Simple Notification Service (Amazon SNS) – Veröffentlichen Sie, falls konfiguriert, detaillierte Nachrichten über Ihren Image-Status in einem SNS-Thema, das Sie abonnieren. Weitere Informationen finden Sie unter Was ist Amazon SNS? im Entwicklerhandbuch für Amazon Simple Notification Service.

Themen zur Produkt- und Serviceintegration

- AWS CloudTrail -Integration in Image Builder
- Amazon- CloudWatch Logs-Integration in Image Builder
- Amazon- EventBridge Integration in Image Builder
- Amazon Inspector-Integration in Image Builder
- AWS Marketplace -Integration in Image Builder
- Amazon SNS-Integration in Image Builder
- Compliance-Produkte für Ihre Image-Builder-Images

AWS CloudTrail -Integration in Image Builder

Dieser Service unterstützt AWS CloudTrail. CloudTrail ist ein Service, der AWS Aufrufe für Ihr aufzeichnet AWS-Konto und Protokolldateien an einen Amazon S3-Bucket übermittelt. Mithilfe der von gesammelten Informationen können Sie feststellen CloudTrail, welche Anforderungen erfolgreich an gestellt wurdenAWS-Services, wer die Anforderung gestellt hat, wann sie gestellt wurde usw. Weitere Informationen zur CloudTrail Integration mit Image Builder finden Sie unter Protokollieren von EC2-Image-Builder-API-Aufrufen mit AWS CloudTrail.

Weitere Informationen zu CloudTrail, einschließlich der Aktivierung und der Suche nach Ihren Protokolldateien, finden Sie im AWS CloudTrail -Benutzerhandbuch.

Amazon- CloudWatch Logs-Integration in Image Builder

CloudWatch Die Protokollunterstützung ist standardmäßig aktiviert. Protokolle werden während des Erstellungsprozesses auf der Instance aufbewahrt und an CloudWatch Protokolle gestreamt. Die Instance-Protokolle werden vor der Image-Erstellung aus der Instance entfernt.

Build-Protokolle werden nach der Gruppe und dem Stream von Image Builder CloudWatch Logs gestreamt:

LogGroup:

```
/aws/imagebuilder/ImageName
```

LogStream (x.x.x/x):

```
ImageVersion/ImageBuildVersion
```

Sie können das CloudWatch Logs-Streaming deaktivieren, indem Sie die folgenden Berechtigungen entfernen, die dem Instance-Profil zugeordnet sind.

```
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:CreateLogGroup",
            "logs:PutLogEvents"
        ],
        "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
    }
]
```

Zur erweiterten Fehlerbehebung können Sie vordefinierte Befehle und Skripts mit AWS Systems Manager Run Command ausführen. Weitere Informationen finden Sie unter Fehlerbehebung bei EC2 Image Builder.

Amazon- EventBridge Integration in Image Builder

Amazon EventBridge ist ein Serverless-Event-Bus-Service, mit dem Sie Ihre Image-Builder-Anwendung mit zugehörigen Daten aus anderen verbinden könnenAWS-Services. In EventBridgegleicht eine Regel eingehende Ereignisse ab und sendet sie zur Verarbeitung an Ziele. Eine einzelne Regel kann ein Ereignis an mehrere Ziele senden, und diese Ereignisse werden dann parallel ausgeführt.

Mit können Sie Ihr automatisieren AWS-Services und automatisch auf Systemereignisse reagieren EventBridge, z. B. bei Problemen mit der Anwendungsverfügbarkeit oder Ressourcenänderungen. Ereignisse von AWS-Services werden nahezu EventBridge in Echtzeit an übermittelt. Sie können Regeln einrichten, die auf eingehende Ereignisse reagieren, um Aktionen zu initiieren, z. B. das Senden eines Ereignisses an eine Lambda-Funktion, wenn sich der Status einer EC2-Instance von ausstehend zu ausgeführt ändert. Diese werden als Muster bezeichnet. Informationen zum Erstellen einer Regel basierend auf einem Ereignismuster finden Sie unter Erstellen von Amazon- EventBridge Regeln, die auf Ereignisse reagieren im Amazon- EventBridge Benutzerhandbuch.

Zu den Aktionen, die automatisch initiiert werden können, gehören die folgenden:

- Aufrufen einer AWS Lambda-Funktion
- Aufrufen eines Amazon-EC2-Ausführungsbefehls
- Weitergabe des Ereignisses an Amazon Kinesis Data Streams
- Aktivieren eines AWS Step Functions-Status-Automaten
- Benachrichtigen eines Amazon SNS-Themas oder einer Amazon SQS-Warteschlange

Sie können auch Planungsregeln für den Standard-Event-Bus einrichten, um in regelmäßigen Abständen eine Aktion auszuführen, z. B. die Ausführung einer Image-Builder-Pipeline zur vierteljährlichen Aktualisierung eines Images. Es gibt zwei Arten von Zeitplanausdrücken:

- Cron-Ausdrücke – Im folgenden Beispiel für einen Cron-Ausdruck wird die Ausführung einer Aufgabe jeden Tag um Mitternacht UTC+0 geplant:

  ```
  cron(0 12 * * ? *)
  ```

  Weitere Informationen zur Verwendung von Cron-Ausdrücken mit EventBridgefinden Sie unter Cron-Ausdrücke im Amazon- EventBridge Benutzerhandbuch.
- Rate-Ausdrücke – Im folgenden Beispiel für einen Rate-Ausdruck wird die Ausführung einer Aufgabe alle 12 Stunden geplant:

  ```
  rate(12 hour)
  ```

  Weitere Informationen zur Verwendung von Rate-Ausdrücken mit EventBridgefinden Sie unter Rate-Ausdrücke im Amazon- EventBridge Benutzerhandbuch.

Weitere Informationen zur EventBridge Integration von in Image-Builder-Image-Pipelines finden Sie unter Verwenden von EventBridge Regeln mit Image-Builder-Pipelines.

Amazon Inspector-Integration in Image Builder

Wenn Sie das Sicherheitsscannen mit Amazon Inspector aktivieren, scannt es kontinuierlich Maschinen-Images und laufende Instances in Ihrem Konto auf Schwachstellen im Betriebssystem und in der Programmiersprache. Wenn diese Option aktiviert ist, erfolgt das Sicherheitsscannen automatisch und Image Builder kann einen Snapshot der Ergebnisse aus Ihrer Test-Instance speichern, wenn Sie ein neues Image erstellen. Amazon Inspector ist ein kostenpflichtiger Service.

Wenn Amazon Inspector Schwachstellen in Ihren Software- oder Netzwerkeinstellungen erkennt, werden die folgenden Aktionen ausgeführt:

- Benachrichtigt Sie, dass ein Ergebnis vorliegt.
- Bewertet den Schweregrad der Erkenntnis. Die Schweregradbewertung kategorisiert Schwachstellen, um Ihnen bei der Priorisierung Ihrer Ergebnisse zu helfen, und enthält die folgenden Werte:
  - Nicht verteilt
  - Informativ
  - Niedrig
  - Medium
  - Hoch
  - Kritisch
- Enthält Informationen über die Erkenntnis und Links zu zusätzlichen Ressourcen für weitere Details.
- Bietet Anleitungen zur Behebung von Problemen, die das Ergebnis generiert haben.

Konfigurieren von Sicherheitsscans

Wenn Sie Amazon Inspector für Ihr Konto aktiviert haben, scannt Amazon Inspector automatisch die EC2-Instances, die Image Builder startet, um ein neues Image zu erstellen und zu testen. Diese Instances haben während des Build- und Testprozesses eine kurze Lebensdauer und ihre Ergebnisse laufen normalerweise ab, sobald diese Instances heruntergefahren werden. Um Sie bei der Untersuchung und Behebung von Erkenntnissen für Ihr neues Image zu unterstützen, kann Image Builder optional alle Erkenntnisse, die Amazon Inspector während des Erstellungsprozesses auf Ihrer Test-Instance identifiziert hat, als Snapshot speichern.

Informationen zum Konfigurieren von Sicherheitsscans für Ihre Pipeline finden Sie unter Konfigurieren von Sicherheitsscans für Image-Builder-Images in der AWS Management Console.

Überprüfen der Sicherheitserkenntnisse

In der Image-Builder-Konsole können Sie Sicherheitserkenntnisse für alle Ihre Image-Builder-Ressourcen an einem Ort anzeigen. Sie können alle Erkenntnisse auf der Seite Sicherheitserkenntnisse im Abschnitt Sicherheitsübersicht sehen oder Ihre Erkenntnisse nach Schwachstellen, Image-Pipeline oder Image gruppieren. Die Konsole zeigt standardmäßig alle

Sicherheitserkenntnisse an. Im Übersichtsbereich für die Option Alle Sicherheitserkenntnisse wird die Anzahl der Erkenntnisse angezeigt, die Sie für jeden Schweregrad haben. Weitere Informationen finden Sie unter Verwalten von Sicherheitsergebnissen für Image-Builder-Images in der AWS Management Console.

Weitere Informationen zu den Erkenntnissen zu Amazon Inspector-Schwachstellen finden Sie unter Erkenntnisse in Amazon Inspector verstehen im Amazon Inspector-Benutzerhandbuch.

AWS Marketplace -Integration in Image Builder

AWS Marketplace ist ein kuratierter digitaler Katalog, in dem Sie Software, Daten und Services von Drittanbietern finden und abonnieren können, die Ihnen helfen, Lösungen zu entwickeln, die Ihren Geschäftsanforderungen entsprechen. AWS Marketplace bietet authentifizierte Käufer und registrierte Verkäufer sowie Softwarelisten aus beliebten Kategorien wie Sicherheit, Netzwerk, Speicher, Machine Learning und mehr.

Bei einem -AWS MarketplaceVerkäufer kann es sich um einen unabhängigen Softwareanbieter (ISV), einen Verkäufer oder eine Person handeln, die etwas zu bieten hat, das mit -AWSProdukten und -Services zusammenarbeitet. Wenn der Verkäufer ein Produkt an AWS Marketplace übermittelt, legt er einen Preis für das Produkt und die Nutzungsbedingungen fest. Käufer stimmen den für das Angebot festgelegten Preisen, Bedingungen und Bedingungen zu. Weitere Informationen zu AWS Marketplacefinden Sie unter Was ist AWS Marketplace?

> **Note**
>
> Datenproduktanbieter müssen die Voraussetzungen für die Berechtigung von AWS Data Exchange erfüllen. Weitere Informationen finden Sie unter Bereitstellen von Datenprodukten im AWS Data Exchange im AWS Data Exchange-Benutzerhandbuch.

AWS Marketplace-Integrationsfunktionen

Image Builder lässt sich in integrierenAWS Marketplace, um die folgenden Funktionen direkt von der Image-Builder-Konsole aus bereitzustellen:

- Suchen Sie nach Bildprodukten, die in verfügbar sindAWS Marketplace.
- Sehen Sie sich eine Liste Ihrer aktuellen AWS Marketplace Produktabonnements an.

- Verwenden Sie ein AWS Marketplace Image-Produkt als Basis-Image für ein Image-Builder-Rezept.

Für Produkte, die zugehörige AWS Task Orchestrator and Executor (AWSTOE)-Komponenten enthalten, können Sie in der Konsole und in der API, dem SDK und der CLI nach dem Produkteigentümer filtern. Weitere Informationen finden Sie unter AWSTOE Komponenten auflisten.

AWS Marketplace Image-Produkte über die Image-Builder-Konsole suchen

Image Builder lässt sich in integrierenAWS Marketplace, um Ihre Image-Produktabonnements direkt aus dem -AWS MarketplaceAbschnitt in der Image-Builder-Konsole anzuzeigen. Sie können auch auf der Seite Image-Produkte nach AWS Marketplace Image-Produkten suchen, ohne die Image-Builder-Konsole zu verlassen.

Gehen Sie folgendermaßen vor, um ein AWS Marketplace Image-Produkt in der Image-Builder-Konsole zu finden:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich im AWS Marketplace Abschnitt Image-Produkte aus.
3. Auf der Seite Image-Produkte finden Sie eine Zusammenfassung der Image-Produkte, die Sie auf der Registerkarte Abonnements abonniert haben, oder Sie können auf der AWS Marketplace Registerkarte nach Image-Produkten suchen.

   Image Builder filtert Produkte von vorab, AWS Marketplace um sich auf Computer-Images zu konzentrieren, die Sie in Ihren Image-Builder-Rezepten verwenden können. Für weitere Informationen zur AWS Marketplace Integration mit Image Builder wählen Sie die Registerkarte aus, die dem entspricht, was Sie sehen möchten.

   AWS Marketplace

   Diese Registerkarte enthält zwei Bereiche. Auf der linken Seite können Sie im Bereich Ergebnisse verfeinern Ihre Ergebnisse filtern, um die Produkte zu finden, die Sie abonnieren möchten. Auf der rechten Seite werden im Bereich Produkte suchen die Produkte angezeigt, die Ihren Filterkriterien entsprechen, und Sie haben auch die Möglichkeit, nach Produktnamen zu suchen.

   Ergebnisse verfeinern

Die folgende Liste zeigt nur einige der Filter, die Sie auf Ihre Produktsuche anwenden können:

- Wählen Sie eine oder mehrere Produktkategorien aus, z. B. Infrastruktursoftware oder Machine Learning.
- Wählen Sie die Betriebssysteme für Ihr Image-Produkt oder alle Produkte für eine bestimmte Betriebssystemplattform aus, z. B. Alle Linux/Unix-.
- Wählen Sie einen oder mehrere Publisher aus, um ihre verfügbaren Produkte anzuzeigen. Wählen Sie den Link Alle anzeigen, um alle Publisher anzuzeigen, die Produkte haben, die den von Ihnen angewendeten Filtern entsprechen.

> **Note**
>
> Publisher-Namen sind nicht in alphabetischer Reihenfolge. Wenn Sie nach einem bestimmten Herausgeber wie suchen`Center for Internet Security`, können Sie einen Teil des Namens in das Suchfeld oben im Dialogfeld Alle Herausgeber eingeben. Sie sollten den Namen als Abkürzung angeben, z. B. führt `CIS` möglicherweise nicht zu den Ergebnissen, nach denen Sie suchen.
> Sie können auch die Seite mit den Herausgebernamen nach Seiten durchsuchen.

Die Filteroptionen sind dynamisch. Jede Auswahl, die Sie treffen, wirkt sich auf Ihre Optionen für alle anderen Kategorien aus. In sind Tausende von Produkten verfügbarAWS Marketplace. Je mehr Sie filtern können, desto wahrscheinlicher ist es, dass Sie das finden, was Sie möchten.

Suche nach Produkten

Um ein bestimmtes Produkt anhand des Namens zu finden, können Sie einen Teil des Namens in die Suchleiste oben in diesem Bereich eingeben. Jedes Produktergebnis enthält die folgenden Details:

- Der Produktname und das Logo. Beide sind mit der Produktdetailseite in verknüpftAWS Marketplace. Die Detailseite wird in einer neuen Registerkarte in Ihrem Browser geöffnet. Von dort aus können Sie das Image-Produkt abonnieren, wenn Sie es in einem Image-Builder-Rezept verwenden möchten. Weitere Informationen finden Sie unter Kaufen von Produkten im AWS Marketplace -Käuferhandbuch.

Wenn Sie das Image-Produkt in abonnierenAWS Marketplace, wechseln Sie zurück zur Registerkarte Image Builder in Ihrem Browser und aktualisieren Sie Ihre Liste der abonnierten Image-Produkte, um sie anzuzeigen.

> **Note**
>
> Es kann einige Minuten dauern, bis Ihr neues Abonnement verfügbar ist.

- Der Name des Herausgebers. Dies ist mit der Detailseite des Herausgebers in verknüpftAWS Marketplace. Die Detailseite des Herausgebers wird in einer neuen Registerkarte in Ihrem Browser geöffnet.
- Die Produktversion.
- Die Bewertung des Produkt-Stars und die direkten Links zum Bewertungsabschnitt der Produktdetailseite in AWS Marketplace. Die Detailseite wird in einer neuen Registerkarte in Ihrem Browser geöffnet.
- Die ersten Zeilen der Produktbeschreibung.

Direkt unter der Suchleiste können Sie sehen, wie viele Ergebnisse Ihre Suche erzeugt hat und welche Teilmenge dieser Ergebnisse derzeit angezeigt wird. Sie können zusätzliche Steuerelemente auf der rechten Seite des Bedienfelds verwenden, um Ihre Einstellungen für die Anzahl der gleichzeitig anzuzeigenden Produkte und die Sortierreihenfolge, die auf Ihre Ergebnisse angewendet werden soll, anzupassen. Sie können die Paginierungssteuerung auch verwenden, um Ihre Ergebnisse zu durchsuchen.

Subscriptions

Auf dieser Registerkarte finden Sie eine Liste der Image-Produkte, die Sie in abonniert habenAWS Marketplace. Jedes abonnierte Produkt zeigt die folgenden Details:

- Der Produktname. Dies ist mit der Produktdetailseite in verknüpftAWS Marketplace. Die Produktdetailseite für Ihr abonniertes Produkt wird in einer neuen Registerkarte in Ihrem Browser geöffnet.
- Der Name des Herausgebers. Dies ist mit der Detailseite des Herausgebers in verknüpftAWS Marketplace. Die Detailseite des Herausgebers wird in einer neuen Registerkarte in Ihrem Browser geöffnet.
- Die Produktversion, die Sie abonniert haben.

- Wenn Ihrem abonnierten Produkt eine zugeordnete Komponente zugeordnet ist, zeigt Image Builder einen Link zu den AWSTOE Komponentendetails an.

Oben auf der Seite können Sie nach einem bestimmten Produkt nach Namen suchen oder Ihre Ergebnisse mit den Paginierungskontrollen durchsuchen. Um ein abonniertes Produkt als Basis-Image für ein neues Rezept zu verwenden, wählen Sie ein abonniertes Produkt aus und klicken Sie auf Neues Rezept erstellen. Image Builder wählt standardmäßig das erste Produkt in Ihrer Liste aus.

> **Note**
>
> Wenn Sie nach einem Produkt suchen, das Sie gerade abonniert haben, und es nicht in der Liste angezeigt wird, verwenden Sie die Schaltfläche Aktualisieren oben auf der Registerkarte, um Ihre Ergebnisse zu aktualisieren. Es kann einige Minuten dauern, bis ein neues Abonnement in der Liste angezeigt wird.

Verwenden eines AWS Marketplace Image-Produkts in Image-Builder-Rezepten

In der Image-Builder-Konsole gibt es zwei Möglichkeiten, ein neues Image-Rezept basierend auf einem Ihrer abonnierten Image-Produkte zu erstellen.

1. Sie können auf der Seite Image-Produkte wie folgt beginnen:

    1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
    2. Wählen Sie im Navigationsbereich im AWS Marketplace Abschnitt Image-Produkte aus.
    3. Öffnen Sie die Registerkarte Abonnements.
    4. Wählen Sie das abonnierte Image-Produkt aus, das Sie als Basis-Image in Ihrem Rezept verwenden möchten.
    5. Wählen Sie Neues Rezept erstellen aus. Dadurch wird die Seite Rezept erstellen geöffnet, auf der die Option AWS Marketplace Bilder und Ihr abonniertes Bildprodukt vorausgewählt sind.
    6. Konfigurieren Sie die verbleibenden Einstellungen für Ihr Rezept wie gewohnt. Weitere Informationen zu Image-Rezepten finden Sie unter Erstellen einer neuen Version eines Image-Rezepts.

2. Sie können auch die Seite Rezept erstellen öffnen und ein AWS Marketplace Image-Produkt auswählen, das Sie als Basis-Image verwenden möchten.

   1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
   2. Wählen Sie im Navigationsbereich Image-Rezepte im AWS Marketplace Abschnitt aus. Dies zeigt Ihnen eine Liste von Image-Rezepten, die Sie erstellt haben.
   3. Wählen Sie Create image recipe (Image-Rezept erstellen) aus. Dadurch wird die Seite Rezept erstellen geöffnet.
   4. Geben Sie Ihren Rezeptnamen und Ihre Version wie gewohnt im Abschnitt Rezeptdetails ein.
   5. Wählen Sie im Abschnitt Basisbild die Option AWS Marketplace Bilder aus. Dadurch wird eine Liste der AWS Marketplace Image-Produkte angezeigt, die Sie auf der Registerkarte Abonnements abonniert haben. Sie können Ihr Basis-Image aus der Liste auswählen.

      Sie können auch AWS Marketplace direkt auf der AWS Marketplace Registerkarte nach anderen Bildprodukten suchen, die in verfügbar sind. Wählen Sie Produkte hinzufügen oder öffnen Sie die AWS Marketplace Registerkarte direkt. Weitere Informationen zum Festlegen von Filtern und Suchen in der finden Sie AWS Marketplaceunter AWS Marketplace Image-Produkte über die Image-Builder-Konsole suchen.
   6. Geben Sie die verbleibenden Details wie gewohnt ein und wählen Sie Rezept erstellen aus.

> **Note**
>
> Wenn Ihr Image-Produktabonnement eine AWSTOE Build-Komponente enthält, können Sie sie aus der Liste Build-Komponenten auswählen. Wählen Sie `Third party managed` aus der Liste der Komponentenbesitzertypen aus, um sie anzuzeigen. Wenn Ihr Produktabonnement eine AWSTOE Testkomponente enthält, befolgen Sie dasselbe Verfahren für die Liste Testkomponenten.

Amazon SNS-Integration in Image Builder

Amazon Simple Notification Service (Amazon SNS) ist ein verwalteter Service, der die asynchrone Nachrichtenzustellung von Publishern an Abonnenten (auch bekannt als Produzenten und Verbraucher) bereitstellt. Sie können ein SNS-Thema in Ihrer Infrastrukturkonfiguration angeben. Wenn Sie ein Image erstellen oder eine Pipeline ausführen, kann Image Builder detaillierte

Nachrichten über Ihren Image-Status zu diesem Thema veröffentlichen. Wenn der Image-Status einen der folgenden Status erreicht, veröffentlicht Image Builder eine Nachricht:

- `AVAILABLE`
- `FAILED`

Ein Beispiel für eine SNS-Nachricht von Image Builder finden Sie unter SNS-Nachrichtenformat. Wenn Sie ein neues SNS-Thema erstellen möchten, finden Sie weitere Informationen unter Erste Schritte mit Amazon SNS im Amazon Simple Notification Service-Entwicklerhandbuch.

Verschlüsselte SNS-Themen

Wenn Ihr SNS-Thema verschlüsselt ist, müssen Sie die Berechtigung in der AWS KMS key Richtlinie für die Image-Builder-Servicerolle erteilen, die folgenden Aktionen auszuführen:

- `kms:Decrypt`
- `kms:GenerateDataKey`

> **Note**
>
> Wenn Ihr SNS-Thema verschlüsselt ist, muss sich der Schlüssel, der dieses Thema verschlüsselt, in dem Konto befinden, in dem der Image-Builder-Service ausgeführt wird. Image Builder kann keine Benachrichtigungen an SNS-Themen senden, die mit Schlüsseln von anderen Konten verschlüsselt sind.

Beispiel für das Hinzufügen einer KMS-Schlüsselrichtlinie

Das folgende Beispiel zeigt den zusätzlichen Abschnitt, den Sie der KMS-Schlüsselrichtlinie hinzufügen. Verwenden Sie den Amazon-Ressourcennamen (ARN) für die serviceverknüpfte IAM-Rolle, die Image Builder unter Ihrem Konto erstellt hat, als Sie zum ersten Mal ein Image-Builder-Image erstellt haben. Weitere Informationen zur serviceverknüpften Rolle von Image Builder finden Sie unter Verwenden von serviceverknüpften Rollen für EC2 Image Builder.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
```

```
      "AWS": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
    },
    "Action": [
      "kms:GenerateDataKey*",
      "kms:Decrypt"
    ],
    "Resource": "*"
  }]
}
```

Sie können eine der folgenden Methoden verwenden, um den ARN abzurufen.

AWS Management Console

Gehen Sie folgendermaßen vor, um den ARN für die serviceverknüpfte Rolle, die Image Builder unter Ihrem Konto erstellt hatAWS Management Console, über die abzurufen:

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Suchen Sie nach `ImageBuilder`und wählen Sie den folgenden Rollennamen aus den Ergebnissen aus: `AWSServiceRoleForImageBuilder`. Dadurch wird die Seite mit den Rollendetails angezeigt.
4. Um den ARN in Ihre Zwischenablage zu kopieren, wählen Sie das Symbol neben dem ARN-Namen.

AWS CLI

Um den ARN für die serviceverknüpfte Rolle abzurufen, die Image Builder unter Ihrem Konto von erstellt hatAWS CLI, verwenden Sie den Befehl IAM get-role wie folgt.

```
aws iam get-role --role-name AWSServiceRoleForImageBuilder
```

Teilweise Beispielausgabe:

```
{
    "Role": {
        "Path": "/aws-service-role/imagebuilder.amazonaws.com/",
        "RoleName": "AWSServiceRoleForImageBuilder",
        ...
```

```
        "Arn": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
        ...
}
```

SNS-Nachrichtenformat

Nachdem Image Builder eine Nachricht in Ihrem Amazon SNS-Thema veröffentlicht hat, können andere Services, die das Thema abonnieren, nach dem Nachrichtenformat filtern und feststellen, ob es die Kriterien für weitere Maßnahmen erfüllt. Beispielsweise kann eine Erfolgsmeldung eine Aufgabe auslösen, um einen -AWS Systems ManagerParameterspeicher zu aktualisieren oder einen externen Compliance-Test-Workflow für das Ausgabe-AMI zu starten.

Das folgende Beispiel zeigt die JSON-Nutzlast für eine typische Nachricht, die Image Builder veröffentlicht, wenn ein Pipeline-Build abgeschlossen wird, und ein Linux-Image erstellt.

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image",
  "semver": 1237940039285380274899124227,
  "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3",
  "name": "example-linux-image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 3,
  "state": {
    "status": "AVAILABLE"
  },
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-linux-
image/1.0.0",
    "name": "amjule-barebones-linux",
    "version": "1.0.0",
    "components": [
      {
        "componentArn": "arn:aws:imagebuilder:us-west-1:123456789012:component/update-
linux/1.0.2/1"
      }
    ],
```

```
    "platform": "Linux",
    "parentImage": "arn:aws:imagebuilder:us-west-1:987654321098:image/amazon-linux-2-
x86/2022.6.14/1",
    "blockDeviceMappings": [
      {
        "deviceName": "/dev/xvda",
        "ebs": {
          "encrypted": false,
          "deleteOnTermination": true,
          "volumeSize": 8,
          "volumeType": "gp2"
        }
      }
    ],
    "dateCreated": "Feb 24, 2021 12:31:54 AM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-
linux-image/1.0.0"
    },
    "workingDirectory": "/tmp",
    "accountId": "462045008730"
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-pipeline/
example-linux-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-configuration/
example-linux-infra-config-uswest1",
    "name": "example-linux-infra-config-uswest1",
    "instanceProfileName": "example-linux-ib-baseline-admin",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-
configuration/example-linux-infra-config-uswest1"
    },
    "logging": {
      "s3Logs": {
        "s3BucketName": "12345-example-linux-testbucket-uswest1"
      }
    },
    "keyPair": "example-linux-key-pair-uswest1",
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-1:123456789012:example-linux-ibnotices-
uswest1",
```

```
    "dateCreated": "Feb 24, 2021 12:31:55 AM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-configuration/
example-linux-distribution",
    "name": "example-linux-distribution",
    "dateCreated": "Feb 24, 2021 12:31:56 AM",
    "distributions": [
      {
        "region": "us-west-1",
        "amiDistributionConfiguration": {}
      }
    ],
    "tags": {
      "internalId": "345abc67-8910-12d3-4ef5-67a8b90c12de",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-
configuration/example-linux-distribution"
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 28, 2022 1:13:45 AM",
  "outputResources": {
    "amis": [
      {
        "region": "us-west-1",
        "image": "ami-01a23bc4def5a6789",
        "name": "example-linux-image 2022-07-28T01-14-17.416Z",
        "accountId": "123456789012"
      }
    ]
  },
  "buildExecutionId": "ab0cd12e-34fa-5678-b901-2c3456d789e0",
  "testExecutionId": "6a7b8901-cdef-234a-56b7-8cd89ef01234",
  "distributionJobId": "1f234567-8abc-9d0e-1234-fa56b7c890de",
  "integrationJobId": "432109b8-afe7-6dc5-4321-0ba98f7654e3",
  "accountId": "123456789012",
  "osVersion": "Amazon Linux 2",
  "enhancedImageMetadataEnabled": true,
  "buildType": "USER_INITIATED",
```

```
  "tags": {
    "internalId": "901e234f-a567-89bc-0123-d4e567f89a01",
    "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3"
  }
}
```

Das folgende Beispiel zeigt die JSON-Nutzlast für eine typische Nachricht, die Image Builder für einen Pipeline-Build-Fehler für ein Linux-Image veröffentlicht.

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image",
  "semver": 12379400392853802748991242 31,
  "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-image/1.0.0/7",
  "name": "My Example Image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 7,
  "state": {
    "status": "FAILED",
    "reason": "Image Failure reason."
  },
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-
image/1.0.0",
    "name": "My Example Image",
    "version": "1.0.0",
    "description": "Testing Image recipe",
    "components": [
      {
        "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-
example-image-component/1.0.0/1"
      }
    ],
    "platform": "Linux",
    "parentImage": "ami-0cd12345db678d90f",
    "dateCreated": "Jun 21, 2022 11:36:14 PM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
example-image/1.0.0"
```

```
    },
    "accountId": "123456789012"
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-
example-image-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/
my-example-infra-config",
    "name": "SNS topic Infra config",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
      "t2.micro"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/my-example-infra-config"
    },
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:example-pipeline-notification-
topic",
    "dateCreated": "Jul 5, 2022 7:31:53 PM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-
example-distribution-config",
    "name": "New distribution config",
    "dateCreated": "Dec 3, 2021 9:24:22 PM",
    "distributions": [
      {
        "region": "us-west-2",
        "amiDistributionConfiguration": {},
        "fastLaunchConfigurations": [
          {
            "enabled": true,
            "snapshotConfiguration": {
              "targetResourceCount": 2
            },
```

```
            "maxParallelLaunches": 2,
            "launchTemplate": {
              "launchTemplateId": "lt-01234567890"
            },
            "accountId": "123456789012"
          }
        ]
      }
    ],
    "tags": {
      "internalId": "1fecd23a-4f56-7f89-01e2-345678abbe90",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/my-example-distribution-config"
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 5, 2022 7:40:15 PM",
  "outputResources": {
    "amis": []
  },
  "accountId": "123456789012",
  "enhancedImageMetadataEnabled": true,
  "buildType": "SCHEDULED",
  "tags": {
    "internalId": "456c78b9-0e12-3f45-afb6-7e89b0f1a23b",
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image/1.0.0/7"
  }
}
```

Compliance-Produkte für Ihre Image-Builder-Images

Angesichts sich ständig weiterentwickelnder Sicherheitsstandards kann es eine Herausforderung sein, die Compliance aufrechtzuerhalten und Ihre Organisation vor Cyberbedrohungen zu schützen. Um sicherzustellen, dass Ihre benutzerdefinierten Images konform sind, und diesen Weg durch automatische Updates zu halten, wenn Publisher neue Versionen veröffentlichen, ist Image Builder in AWS Marketplace Compliance-Produkte und -AWSTOEKomponenten integriert.

Image Builder lässt sich in die folgenden Compliance-Produkte integrieren:

- Center for Internet Security (CIS) Benchmarks – Hardening

Sie können CIS-gehärtete Images und die zugehörigen CIS-Hardening-Komponenten verwenden, um benutzerdefinierte Images zu erstellen, die den neuesten CIS Benchmarks Level 1-Richtlinien entsprechen. CIS-gehärtete Images sind in verfügbarAWS Marketplace. Weitere Informationen zum Einrichten und Verwenden von CIS-gehärteten Images und -Hardening-Komponenten finden Sie in den Schnellstartanleitungen im CIS-Website-Supportportal.

> **Note**
>
> Wenn Sie ein CIS-gehärtetes Image abonnieren, erhalten Sie auch Zugriff auf die zugehörige Build-Komponente, die ein Skript ausführt, um CIS-Benchmark-Level-1-Richtlinien für Ihre Konfiguration durchzusetzen. Weitere Informationen finden Sie unter CIS-Harding-Komponenten.

- Security Technical Implementation Guides (STIG)

  Für STIG-Compliance kann die von Amazon verwaltete AWS Task Orchestrator and Executor (AWSTOE) STIG-Komponenten in Ihren Image-Builder-Rezepten verwenden. STIG-Komponenten scannen Ihre Build-Instance auf Fehlkonfigurationen und führen ein Korrekturskript aus, um Probleme zu beheben, die sie finden. Wir können die STIG-Compliance für die Images, die Sie mit Image Builder erstellen, nicht garantieren. Sie müssen mit dem Compliance-Team Ihrer Organisation zusammenarbeiten, um zu überprüfen, ob Ihr endgültiges Image konform ist. Eine vollständige Liste der AWSTOE STIG-Komponenten, die Sie in Ihren Image-Builder-Rezepten verwenden können, finden Sie unter Von Amazon verwaltete STIG-Harding-Komponenten für EC2 Image Builder.

Überwachen von Ereignissen und Protokollen in EC2 Image Builder

Um die Zuverlässigkeit, Verfügbarkeit und Leistung Ihrer EC2 Image Builder-Pipelines aufrechtzuerhalten, ist es wichtig, Ereignisse und Protokolle zu überwachen. Ereignisse und Protokolle helfen Ihnen, das große Bild zu sehen und sich mit den Details vertraut zu machen, wenn ein API-Aufruf fehlschlägt. Image Builder lässt sich in Services integrieren, die Warnungen senden und automatisierte Antworten auslösen können, wenn Ereignisse den von Ihnen konfigurierten Kriterien entsprechen.

In den folgenden Themen werden Überwachungstechniken beschrieben, die Sie über Services verwenden können, die in Image Builder integriert sind.

Überwachen von Ereignissen und Protokollen

- Protokollieren von EC2-Image-Builder-API-Aufrufen mit AWS CloudTrail

Protokollieren von EC2-Image-Builder-API-Aufrufen mit AWS CloudTrail

EC2 Image Builder ist in integriert, einem ServiceAWS CloudTrail, der die Aktionen aller API-Aufrufe für von einem Benutzer, einer Rolle oder einem -AWSService über die Image-Builder-API protokolliert. CloudTrail erfasst Image Builder als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Image-Builder-Konsole und Codeaufrufe der Image-Builder-API-Operationen.

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen S3-Bucket aktivieren, einschließlich Ereignissen für Image Builder. 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 Image Builder 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 CloudTrailfinden Sie im AWS CloudTrail -Benutzerhandbuch.

Image-Builder-Informationen in CloudTrail

CloudTrail wird auf Ihrem aktiviertAWS-Konto, wenn Sie das Konto erstellen. Wenn eine Aktivität in Image Builder auftritt, wird diese Aktivität in einem CloudTrail Ereignis zusammen mit anderen -AWSServiceereignissen im Ereignisverlauf aufgezeichnet. Sie können in Ihrem AWS-Konto die neusten Ereignisse anzeigen, suchen und herunterladen. 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 Image Builder, einen Trail. Ein Trail ermöglicht CloudTrail die Bereitstellung von Protokolldateien an einen S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser für alle AWS-Regionen-Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien für den von Ihnen angegebenen S3 Bucket bereit. Darüber hinaus können Sie andere konfigurieren, AWS-Services um die in den CloudTrail Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu agieren. Weitere Informationen finden Sie hier:

- Übersicht zum Erstellen eines Trails.
- CloudTrail Von unterstützte Services und Integrationen .
- Konfigurieren von Amazon SNS-Benachrichtigungen für CloudTrail.
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen.
- Empfangen von CloudTrail Protokolldateien von mehreren Konten.

CloudTrail protokolliert alle Image-Builder-Aktionen, die in der EC2-Image-Builder-API-Referenz dokumentiert sind. Aufrufe der `StartImagePipelineExecution` Aktionen `CreateImagePipeline`, `UpdateInfrastructureConfiguration`und erzeugen beispielsweise Einträge in den CloudTrail Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Gibt an, ob die Anforderung mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anforderung aus einem anderen AWS-Service gesendet wurde

Weitere Informationen darüber, wer ein Ereignis angefordert hat, finden Sie im CloudTrail userIdentity-Element .

Sicherheit in EC2 Image Builder

Die Sicherheit in der Cloud hat für AWS höchste Priorität. Als AWS-Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die zur Erfüllung der Anforderungen von Organisationen entwickelt wurden, für die Sicherheit eine kritische Bedeutung hat.

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. Weitere Informationen zu den Compliance-Programmen, die für EC2 Image Builder gelten, finden Sie unter AWS-Services im Geltungsbereich nach Compliance-Programm-
- Sicherheit in der Cloud – Ihr Verantwortungsumfang wird durch den AWS-Dienst bestimmt, den Sie verwenden. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der geteilten Verantwortung bei der Verwendung von Image Builder einsetzen können. Die folgenden Themen zeigen Ihnen, wie Sie Image Builder konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere verwendenAWS-Services, die Ihnen bei der Überwachung und Sicherung Ihrer Image-Builder-Ressourcen helfen.

Themen

Datenschutz in EC2 Image Builder

Das Modell der AWS geteilten gilt für den Datenschutz in EC2 Image Builder. https://aws.amazon.com/compliance/shared-responsibility-model/ Wie in diesem Modell beschrieben, ist AWS für den Schutz der globalen Infrastruktur verantwortlich, in der die gesamte AWS Cloud ausgeführt wird. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS-Modell der geteilten Verantwortung und in der DSGVO im AWS-Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, AWS-Konto-Anmeldeinformationen zu schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einzurichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor Authentifizierung (MFA).
- Verwenden Sie SSL/TLS für die Kommunikation mit AWS-Ressourcen. Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit AWS CloudTrail ein.
- Verwenden Sie AWS-Verschlüsselungslösungen zusammen mit allen Standardsicherheitskontrollen in AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff auf AWS über eine Befehlszeilenschnittstelle oder über 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.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit Image Builder oder anderen AWS-Services über die Konsole, APIAWS CLI, oder AWS SDKs arbeiten. Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen

Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Verschlüsselung und Schlüsselverwaltung in EC2 Image Builder

Image Builder verschlüsselt Daten während der Übertragung und im Ruhezustand standardmäßig mit einem serviceeigenen KMS-Schlüssel, mit Ausnahme der folgenden:

- Benutzerdefinierte Komponenten – Image Builder verschlüsselt benutzerdefinierte Komponenten mit Ihrem Standard-KMS-Schlüssel oder einem serviceeigenen KMS-Schlüssel.
- Image-Workflows – Image Builder kann Ihre Image-Workflows mit einem vom Kunden verwalteten Schlüssel verschlüsseln, wenn Sie den Schlüssel während der Workflow-Erstellung angeben. Image Builder übernimmt die Verschlüsselung und Entschlüsselung mit Ihrem -Schlüssel, um die Workflows auszuführen, die Sie für Ihre Images konfiguriert haben.

Sie können Ihre eigenen Schlüssel über verwaltenAWS KMS. Sie sind jedoch nicht berechtigt, den Image-Builder-KMS-Schlüssel zu verwalten, der Image Builder gehört. Weitere Informationen zum Verwalten Ihrer KMS-Schlüssel mit finden Sie AWS Key Management Serviceunter Erste Schritte im -AWS Key Management ServiceEntwicklerhandbuch.

Verschlüsselungskontext

Um eine zusätzliche Integritäts- und Authentizitätsprüfung für Ihre verschlüsselten Daten durchzuführen, haben Sie die Möglichkeit, einen Verschlüsselungskontext einzuschließen, wenn Sie die Daten verschlüsseln. Wenn eine Ressource mit einem AWS KMS Verschlüsselungskontext verschlüsselt ist, bindet den Kontext kryptografisch an den Geheimtext. Die Ressource kann nur entschlüsselt werden, wenn der Anforderer eine exakte Übereinstimmung zwischen Groß- und Kleinschreibung für den Kontext bereitstellt.

Die Richtlinienbeispiele in diesem Abschnitt verwenden einen Verschlüsselungskontext, der dem Amazon-Ressourcennamen (ARN) einer Image-Builder-Workflow-Ressource ähnelt.

Verschlüsseln von Image-Workflows mit einem vom Kunden verwalteten Schlüssel

Um eine Schutzebene hinzuzufügen, können Sie Ihre Image-Builder-Workflow-Ressourcen mit Ihrem eigenen kundenverwalteten Schlüssel verschlüsseln. Wenn Sie Ihren vom Kunden verwalteten Schlüssel verwenden, um die von Ihnen erstellten Image-Builder-Workflows zu verschlüsseln, müssen Sie in der Schlüsselrichtlinie Zugriff gewähren, damit Image Builder Ihren Schlüssel beim Verschlüsseln und Entschlüsseln von Workflow-Ressourcen verwenden kann. Sie können den Zugriff

jederzeit widerrufen. Image Builder hat jedoch keinen Zugriff auf Workflows, die bereits verschlüsselt sind, wenn Sie den Zugriff auf den Schlüssel widerrufen.

Der Prozess zum Gewähren von Image Builder-Zugriff zur Verwendung Ihres vom Kunden verwalteten Schlüssels besteht aus zwei Schritten:

Schritt 1: Hinzufügen von Schlüsselrichtlinienberechtigungen für Image-Builder-Workflows

Damit Image Builder Workflow-Ressourcen beim Erstellen oder Verwenden dieser Workflows ver- und entschlüsseln kann, müssen Sie Berechtigungen in der KMS-Schlüsselrichtlinie angeben.

Diese Beispiel-Schlüsselrichtlinie gewährt den Image-Builder-Pipelines Zugriff auf die Verschlüsselung von Workflow-Ressourcen während des Erstellungsprozesses und die Entschlüsselung von Workflow-Ressourcen, um sie zu verwenden. Die Richtlinie gewährt Schlüsseladministratoren auch Zugriff. Der Verschlüsselungskontext und die Ressourcenspezifikation verwenden einen Platzhalter, um alle Regionen abzudecken, in denen Sie über Workflow-Ressourcen verfügen.

Als Voraussetzung für die Verwendung von Image-Workflows haben Sie eine IAM-Workflow-Ausführungsrolle erstellt, die Image Builder die Berechtigung erteilt, Workflow-Aktionen auszuführen. Der Prinzipal für die erste Anweisung, die im Beispiel für die Schlüsselrichtlinie hier gezeigt wird, muss Ihre IAM-Workflow-Ausführungsrolle angeben.

Weitere Informationen zu vom Kunden verwalteten Schlüsseln finden Sie unter Verwalten des Zugriffs auf vom Kunden verwaltete Schlüssel im AWS Key Management Service - Entwicklerhandbuch.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Allow access to build images with encrypted workflow",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/YourImageBuilderExecutionRole"
   },
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "*",
```

```
    "Condition": {
     "StringLike": {
      "kms:EncryptionContext:aws:imagebuilder:arn":
 "arn:aws:imagebuilder:*:111122223333:workflow/*"
     }
    }
   },
   {
    "Sid": "Allow access for key administrators",
    "Effect": "Allow",
    "Principal": {
     "AWS": "arn:aws:iam::111122223333:root"
    },
    "Action": [
     "kms:*"
    ],
    "Resource": "arn:aws:kms:*:111122223333:key/"
   }
  ]
}
```

Schritt 2: Gewähren von Schlüsselzugriff auf Ihre Workflow-Ausführungsrolle

Die IAM-Rolle, die Image Builder zum Ausführen Ihrer Workflows übernimmt, benötigt die Berechtigung, Ihren vom Kunden verwalteten Schlüssel zu verwenden. Ohne Zugriff auf Ihren Schlüssel kann Image Builder Ihre Workflow-Ressourcen damit nicht ver- oder entschlüsseln.

Bearbeiten Sie die Richtlinie für Ihre Workflow-Ausführungsrolle, um die folgende Richtlinienanweisung hinzuzufügen.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Allow access to the workflow key",
   "Effect": "Allow",
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "arn:aws:kms:us-west-2:111122223333:key/key_ID",
   "Condition": {
    "StringLike": {
```

```
      "kms:EncryptionContext:aws:imagebuilder:arn":
  "arn:aws:imagebuilder:*:111122223333:workflow/*"
     }
    }
   }
  ]
}
```

AWS CloudTrail -Ereignisse für Image-Workflows

Die folgenden Beispiele zeigen typische AWS CloudTrail Einträge für die Ver- und Entschlüsselung von Image-Workflows, die mit einem vom Kunden verwalteten Schlüssel gespeichert werden.

Beispiel: GenerateDataKey

Dieses Beispiel zeigt, wie ein CloudTrail Ereignis aussehen könnte, wenn Image Builder die AWS KMS GenerateDataKey API-Aktion aus der Image-BuilderCreateWorkflow-API-Aktion aufruft. Image Builder muss einen neuen Workflow verschlüsseln, bevor die Workflow-Ressource erstellt wird.

```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
   }
  },
  "invokedBy": "imagebuilder.amazonaws.com"
 },
```

```
 "eventTime": "2023-11-21T20:31:03Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "GenerateDataKey",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "key value"
  },
  "keyId": "arn:aws:kms:us-west-2:111122223333:alias/ExampleKMSKey",
  "numberOfBytes": 32
 },
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
   "type": "AWS::KMS::Key",
   "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
  }
 ],
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "111122223333",
 "eventCategory": "Management"
}
```


Beispiel: Entschlüsseln

Dieses Beispiel zeigt, wie ein CloudTrail Ereignis aussehen könnte, wenn Image Builder die AWS KMS Decrypt API-Aktion aus der Image-Builder-GetWorkflowAPI-Aktion aufruft. Image-Builder-Pipelines müssen eine Workflow-Ressource entschlüsseln, bevor sie sie sie verwenden können.


```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
```

```
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
   }
  },
  "invokedBy": "imagebuilder.amazonaws.com"
 },
 "eventTime": "2023-11-21T20:34:25Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "Decrypt",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
  "keyId": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLEzzzzz",
  "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "ABC123def4567890abc12345678/90dE/F123abcDEF+4567890abc123D
+ef1=="
  }
 },
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
   "type": "AWS::KMS::Key",
```

```
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
   }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}
```


Datenspeicherung in EC2 Image Builder

Image Builder speichert keines Ihrer Protokolle im Service. Alle Protokolle werden auf Ihrer Amazon EC2-Instance gespeichert, die zum Erstellen des Images verwendet wird, oder in Ihren Systems-Manager-Automatisierungsprotokollen.

Datenschutz zwischen Netzwerken in EC2 Image Builder

Verbindungen zwischen Image Builder und On-Premises-Standorten, zwischen AZs innerhalb einer -AWSRegion und zwischen -AWSRegionen werden über HTTPS gesichert. Es gibt keine direkten Verbindungen zwischen Konten.

Identity and Access Management für EC2 Image Builder

Themen

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, unterscheidet sich je nach Ihrer Arbeit in Image Builder.

Service-Benutzer – Wenn Sie den Image-Builder-Service zur Ausführung von Aufgaben verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen bereit, die Sie benötigen. Wenn Sie für Ihre Arbeit weitere Image-Builder-Funktionen ausführen, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Featuresweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Wenn Sie nicht auf ein Feature in Image Builder zugreifen können, finden Sie weitere Informationen unter Fehlerbehebung für EC2 Image Builder-Identität und -Zugriff.

Service-Administrator – Wenn Sie in Ihrem Unternehmen für Image-Builder-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollständigen Zugriff auf Image Builder. Ihre Aufgabe besteht darin, zu bestimmen, auf welche Image-Builder-Funktionen und -Ressourcen Ihre Service-Benutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen dazu, wie Ihr Unternehmen IAM mit Image Builder verwenden kann, finden Sie unter Funktionsweise von EC2 Image Builder mit IAM.

IAM-Administrator – Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf Image Builder verfassen können. Beispiele für identitätsbasierte Image-Builder-Richtlinien, die Sie in IAM verwenden können, finden Sie unter Identitätsbasierte Image-Builder-Richtlinien.

Authentifizierung mit Identitäten

Ausführliche Informationen zur Bereitstellung der Authentifizierung für Personen und Prozesse in Ihrem AWS-Kontofinden Sie unter Identitäten im IAM-Benutzerhandbuch.

Funktionsweise von EC2 Image Builder mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf Image Builder zu verwalten, erfahren Sie, welche IAM-Funktionen Sie mit Image Builder verwenden können.

Einen Überblick über das Zusammenwirken von Image Builder und anderen -AWSServices mit den meisten IAM-Funktionen finden Sie unter -AWSServices, die mit IAM funktionieren im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien für Image Builder

Unterstützt Richtlinien auf Identitätsbasis.	Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für Image Builder

Beispiele für identitätsbasierte Image-Builder-Richtlinien finden Sie unter Identitätsbasierte Image-Builder-Richtlinien.

Ressourcenbasierte Richtlinien in Image Builder

Unterstützt ressourcenbasierte Richtlinien	Nein

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Prinzipale können Konten, Benutzer, Rollen, Verbundbenutzer oder AWS-Services umfassen.

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource in unterschiedlichen AWS-Konten befinden, muss ein IAM-Administrator im vertrauenswürdigen Konto auch der Prinzipalentität (Benutzer oder Rolle) die Berechtigung zum Zugriff auf die Ressource

erteilen. Sie erteilen Berechtigungen, indem Sie der juristischen Stelle eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie unter Wie sich IAM-Rollen von ressourcenbasierten Richtlinien unterscheiden im IAM-Benutzerhandbuch.

Richtlinienaktionen für Image Builder

Unterstützt Richtlinienaktionen	Ja

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie die zugehörige AWS-API-Operation. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keine passende API-Operation gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Eine Liste der Image-Builder-Aktionen finden Sie unter Von EC2 Image Builder definierte Aktionen in der Service-Autorisierungs-Referenz.

Richtlinienaktionen in Image Builder verwenden das folgende Präfix vor der Aktion:

```
imagebuilder
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
      "imagebuilder:action1",
      "imagebuilder:action2"
         ]
```

Beispiele für identitätsbasierte Image-Builder-Richtlinien finden Sie unter Identitätsbasierte Image-Builder-Richtlinien.

Richtlinienressourcen für Image Builder

Unterstützt Richtlinienressourcen	Ja

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der Image-Builder-Ressourcentypen und ihrer ARNs finden Sie unter Von EC2 Image Builder definierte Ressourcen in der Service-Autorisierungs-Referenz. Informationen zu den Aktionen, mit denen Sie den ARN einzelner Ressourcen angeben können, finden Sie unter Von EC2 Image Builder definierte Aktionen.

Beispiele für identitätsbasierte Image-Builder-Richtlinien finden Sie unter Identitätsbasierte Image-Builder-Richtlinien.

Richtlinienbedingungsschlüssel für Image Builder

Unterstützt servicespezifische Richtlinienbedingungsschlüssel	Ja

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition` block) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen `AND`-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, wertet AWS die Bedingung mittels einer logischen `OR`-Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und servicespezifische Bedingungsschlüssel. Eine Liste aller globalen AWS-Bedingungsschlüssel finden Sie unter Globale AWS-Bedingungskontextschlüssel im IAM-Benutzerhandbuch.

Eine Liste der Image-Builder-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für EC2 Image Builder in der Service-Autorisierungs-Referenz. Informationen dazu, mit welchen Aktionen und Ressourcen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von EC2 Image Builder definierte Aktionen.

Beispiele für identitätsbasierte Image-Builder-Richtlinien finden Sie unter Identitätsbasierte Image-Builder-Richtlinien.

ACLs in Image Builder

Unterstützt ACLs	Nein

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

ABAC mit Image Builder

Unterstützt ABAC (Tags in Richtlinien)	Teilweise

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In AWS werden diese Attribute als Tags bezeichnet. Sie können Tags an IAM-Entitäten (Benutzer oder Rollen) und mehrere AWS-Ressourcen anfügen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Was ist ABAC? im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

Verwenden temporärer Anmeldeinformationen mit Image Builder

Unterstützt temporäre Anmeldeinformationen	Ja

Einige AWS-Services Featureieren nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, unter anderem darüber, welche AWS-Services mit temporären Anmeldeinformationen arbeiten, finden Sie unter AWS-Services, die mit IAM arbeiten im IAM-Benutzerhandbuch.

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen Methode als einem Benutzernamen und einem Passwort bei der AWS Management Console anmelden. Wenn

Sie beispielsweise über den Single Sign-On (SSO)-Link Ihres Unternehmens auf AWS zugreifen, erstellt dieser Prozess automatisch temporäre Anmeldeinformationen. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Wechseln von Rollen finden Sie unter Wechseln zu einer Rolle (Konsole) im IAM-Benutzerhandbuch.

Sie können mithilfe der AWS CLI- oder AWS-API manuell temporäre Anmeldeinformationen erstellen. Sie können dann diese temporären Anmeldeinformationen verwenden, um auf AWS zuzugreifen. AWS empfiehlt, dass Sie temporäre Anmeldeinformationen dynamisch generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen in IAM.

Serviceübergreifende Prinzipal-Berechtigungen für Image Builder

Unterstützt Forward Access Sessions (FAS)	Ja

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle zum Ausführen von Aktionen in AWS verwenden, gelten Sie als Prinzipal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service auslösen. FAS verwendet die Berechtigungen des Prinzipals, der einen AWS-Service aufruft, in Kombination mit der Anforderung an den AWS-Service, Anforderungen an nachgelagerte Services zu stellen. FAS-Anfragen werden nur dann gestellt, wenn ein Service eine Anfrage erhält, die eine Interaktion mit anderen AWS-Services oder -Ressourcen erfordert. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für Image Builder

Unterstützt Servicerollen	Ja

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Das Ändern der Berechtigungen für eine Servicerolle könnte die Image-Builder-Funktionalität beeinträchtigen. Bearbeiten Sie Servicerollen nur, wenn Image Builder dazu Anleitungen gibt.

Serviceverknüpfte Rollen für Image Builder

Unterstützt serviceverknüpfte Rollen	Nein

Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einem AWS-Service verknüpft ist. Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem AWS-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Weitere Informationen zur serviceverknüpften Rolle von Image Builder finden Sie unter Verwenden von serviceverknüpften Rollen für EC2 Image Builder.

Identitätsbasierte Image-Builder-Richtlinien

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Image Builder unterstützt bestimmte Aktionen, Ressourcen und Bedingungsschlüssel. Informationen zu allen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon EC2 Image Builder im IAM-Benutzerhandbuch.

Aktionen

Richtlinienaktionen in Image Builder verwenden das folgende Präfix vor der Aktion: `imagebuilder:`. Richtlinienanweisungen müssen entweder ein `Action`- oder ein `NotAction`-Element enthalten. Image Builder definiert einen eigenen Satz von Aktionen, die Aufgaben beschreiben, die Sie mit diesem Service durchführen können.

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie wie folgt durch Kommata:

```
"Action": [
      "imagebuilder:action1",
```

```
        "imagebuilder:action2"
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `List` beginnen, einschließlich der folgenden Aktion:

```
"Action": "imagebuilder:List*"
```

Eine Liste der Image-Builder-Aktionen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS-Services im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Ausführliche Informationen zum Verwalten des Zugriffs in durch Erstellen AWS von Richtlinien und Anfügen an IAM-Identitäten oder -AWSRessourcen finden Sie unter Richtlinien und Berechtigungen im IAM-Benutzerhandbuch.

Die IAM-Rolle, die Sie Ihrem Instance-Profil zuordnen, muss über Berechtigungen zum Ausführen der Build- und Testkomponenten verfügen, die in Ihrem Image enthalten sind. Die folgenden IAM-Rollenrichtlinien müssen an die IAM-Rolle angehängt werden, die dem Instance-Profil zugeordnet ist:

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- AmazonSSMManagedInstanceCore

Ressourcen

Administratoren können mit AWS-JSON-Richtlinien festlegen, welche Personen zum Zugriff auf welche Ressourcen berechtigt sind. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Die Image-Builder-Instance-Ressource hat den folgenden Amazon-Ressourcennamen (ARN).

```
arn:aws:imagebuilder:region:account-id:resource:resource-id
```

Weitere Informationen zum Format von ARNs finden Sie unter Amazon-Ressourcennamen (ARNs) und AWS-Service-Namespaces.

Um beispielsweise die `i-1234567890abcdef0` Instance in Ihrer Anweisung anzugeben, verwenden Sie den folgenden ARN.

```
"Resource": "arn:aws:imagebuilder:us-east-1:123456789012:instance/i-1234567890abcdef0"
```

Um alle Instances anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie den Platzhalter (*).

```
"Resource": "arn:aws:imagebuilder:us-east-1:123456789012:instance/*"
```

Einige Image Builder-Aktionen, z. B. das Erstellen von Ressourcen, können nicht für eine bestimmte Ressource ausgeführt werden. In diesen Fällen müssen Sie den Platzhalter (*) verwenden.

```
"Resource": "*"
```

Viele API-Aktionen von EC2 Image Builder umfassen mehrere Ressourcen. Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, trennen Sie die ARNs durch Kommata voneinander.

```
"Resource": [
      "resource1",
      "resource2"
```

Bedingungsschlüssel

Image Builder bietet servicespezifische Bedingungsschlüssel und unterstützt die Verwendung einiger globaler Bedingungsschlüssel. Eine Liste aller globalen AWS-Bedingungsschlüssel finden Sie unter Globale AWS-Bedingungskontextschlüssel im IAM-Benutzerhandbuch. Die folgenden servicespezifischen Bedingungsschlüssel werden bereitgestellt.

Image Builder:CreatedResourceTagKeys

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach dem Vorhandensein von Tag-Schlüsseln in der Anforderung zu filtern. Auf diese Weise können Sie die Ressourcen verwalten, die Image Builder erstellt.

Verfügbarkeit – Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

Image Builder:CreatedResourceTag/<key>

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach den Tag-Schlüssel-Wert-Paaren zu filtern, die an die Ressource angehängt sind, die Image Builder erstellt hat. Auf diese Weise können Sie Image-Builder-Ressourcen über definierte Tags verwalten.

Verfügbarkeit – Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

Image Builder:Ec2MetadataHttpTokens

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach der HTTP-Token-Anforderung für EC2-Instance-Metadaten zu filtern, die in der Anforderung angegeben ist.

Dieser Wert für diesen Schlüssel kann entweder `optional` oder sein`required`.

Verfügbarkeit – Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

Image Builder:StatusTopicArn

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach dem SNS-Themen-ARN in der Anforderung zu filtern, an die Terminalstatusbenachrichtigungen veröffentlicht werden.

Verfügbarkeit – Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

Beispiele

Beispiele für identitätsbasierte Image-Builder-Richtlinien finden Sie unter Identitätsbasierte EC2 Image Builder-Richtlinien.

Ressourcenbasierte Image-Builder-Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die angeben, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für die Image-Builder-Ressource ausführen kann. Image Builder unterstützt ressourcenbasierte Berechtigungsrichtlinien für Komponenten, Images und Image-Rezepte. Ressourcenbasierte Richtlinien ermöglichen die Erteilung von Nutzungsberechtigungen für andere -Konten pro Ressource. Sie können auch eine ressourcenbasierte Richtlinie verwenden, um einem -AWSService den Zugriff auf Ihre Komponenten, Images und Image-Rezepte zu ermöglichen.

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource in unterschiedlichen befindenAWS-Konten, müssen Sie der Prinzipal-Entität auch die Berechtigung für den Zugriff auf die Ressource erteilen. Sie erteilen Berechtigungen, indem Sie der Entität eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie unter Wie sich IAM-Rollen von ressourcenbasierten Richtlinien unterscheiden im IAM-Benutzerhandbuch.

Informationen zum Anfügen einer ressourcenbasierten Richtlinie an eine Komponente, ein Image oder ein Image-Rezept finden Sie unter EC2 Image Builder-Ressourcen freigeben.

> **Note**
>
> Wenn Sie eine Ressourcenrichtlinie mit Image Builder aktualisieren, wird das Update in der RAM-Konsole angezeigt.

Autorisierung basierend auf Image-Builder-Tags

Sie können Tags an Image-Builder-Ressourcen anfügen oder Tags in einer Anforderung an Image Builder übergeben. Um den Zugriff auf der Grundlage von Tags zu steuern, geben

Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `imagebuilder:ResourceTag/key-name`, `aws:RequestTag/key-name`, oder Bedingung `aws:TagKeys` verwenden. Weitere Informationen zum Markieren von Image-Builder-Ressourcen finden Sie unter Markieren einer Ressource (AWS CLI).

IAM-Rollen für Image Builder

Eine IAM-Rolle ist eine Entität in Ihrem AWS-Konto mit spezifischen Berechtigungen.

Verwenden temporärer Anmeldeinformationen mit Image Builder

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Sie erhalten temporäre Sicherheitsanmeldeinformationen, indem Sie AWS STS -API-Operationen wie AssumeRole oder aufrufenGetFederationToken.

Service-verknüpfte Rollen

Serviceverknüpfte Rollen erlauben AWS-Services den Zugriff auf Ressourcen in anderen Services, um eine Aktion in Ihrem Auftrag auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein Benutzer mit Administratorzugriff kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, aber nicht bearbeiten.

Image Builder unterstützt serviceverknüpfte Rollen. Informationen zum Erstellen oder Verwalten von serviceverknüpften Image-Builder-Rollen finden Sie unter Verwenden von serviceverknüpften Rollen für EC2 Image Builder.

Servicerollen

Dieses Feature ermöglicht einem Service das Annehmen einer Servicerolle in Ihrem Namen. Diese Rolle gewährt dem Service Zugriff auf Ressourcen in anderen Diensten, um eine Aktion in Ihrem Namen auszuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Das bedeutet, dass ein -Benutzer mit Administratorzugriff die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Dienstes beeinträchtigen.

Identitätsbasierte EC2 Image Builder-Richtlinien

Themen

- Bewährte Methoden für identitätsbasierte Richtlinien
- Verwenden der Image-Builder-Konsole

Bewährte Methoden für identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand Image-Builder-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Erste Schritte mit AWS-verwaltete Richtlinien und Umstellung auf Berechtigungen mit den geringsten Berechtigungen:Um Ihren Benutzern und Workloads Berechtigungen zu gewähren, verwenden Sie die AWS-verwaltete Richtlinien die Berechtigungen für viele allgemeine Anwendungsfälle gewähren. Sie sind in Ihrem AWS-Konto verfügbar. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom Kunden verwaltete AWS-Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS-verwaltete Richtlinien oder AWS-verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten:Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs:Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Service-Aktionen zu gewähren, wenn diese durch ein bestimmtes AWS-Service, wie beispielsweise AWS CloudFormation, verwendet werden. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten:IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung zum IAM Access Analyzer im IAM-Benutzerhandbuch.

- Bedarf einer Multi-Faktor-Authentifizierung (MFA):Wenn Sie ein Szenario haben, das IAM-Benutzer oder Root-Benutzer in Ihrem AWS-Konto erfordert, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der Image-Builder-Konsole

Um auf die EC2 Image Builder-Konsole zugreifen zu können, müssen Sie über einen Mindestsatz von Berechtigungen verfügen. Mit diesen Berechtigungen können Sie Details zu den Image-Builder-Ressourcen in Ihrem auflisten und anzeigenAWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (IAM-Benutzer oder -Rollen) mit dieser Richtlinie.

Um sicherzustellen, dass Ihre IAM-Entitäten die Image-Builder-Konsole verwenden können, müssen Sie ihnen eine der folgenden AWS verwalteten Richtlinien anfügen:

- AWSImageBuilderReadOnlyAccess-Richtlinie
- AWSImageBuilderFullAccess-Richtlinie

Weitere Informationen zu von Image Builder verwalteten Richtlinien finden Sie unter Verwenden von verwalteten Richtlinien für EC2 Image Builder.

> **Important**
>
> Die AWSImageBuilderFullAccess Richtlinie ist erforderlich, um die serviceverknüpfte Rolle von Image Builder zu erstellen. Wenn Sie diese Richtlinie an eine IAM-Entität anfügen, müssen Sie auch die folgende benutzerdefinierte Richtlinie anfügen und die Ressourcen, die Sie verwenden möchten und die nicht `imagebuilder` im Ressourcennamen enthalten:

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "sns topic arn"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile"
            ],
            "Resource": "instance profile role arn"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "instance profile role arn",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "bucket arn"
        }
    ]
}
```

Für Benutzer, die nur Aufrufe an die AWS CLI oder die AWS API durchführen, müssen Sie keine Mindestberechtigungen in der Konsole erteilen. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die Sie ausführen möchten.

Ressourcenbasierte EC2 Image Builder-Richtlinien

Informationen zum Erstellen einer Komponente finden Sie unter Verwalten von Komponenten mit Image Builder.

Beschränken des Image-Builder-Komponentenzugriffs auf bestimmte IP-Adressen

Im folgenden Beispiel werden jedem Benutzer Berechtigungen zum Ausführen von Image-Builder-Operationen für Komponenten erteilt. Die Anfrage muss jedoch aus dem in der Bedingung angegebenen IP-Adressbereich stammen.

Die Bedingung in dieser Anweisung identifiziert den Bereich 54.240.143.* als zulässigen Bereich für Internetprotokoll 4-Adressen (IPv4-Adressen), mit einer Ausnahme: 54.240.143.188.

Der `Condition`-Block verwendet die Bedingungen `IpAddress` und `NotIpAddress` sowie den Bedingungsschlüssel `aws:SourceIp`, wobei es sich um einen AWS-weiten Bedingungsschlüssel handelt. Weitere Informationen zu diesen Bedingungsschlüsseln finden Sie unter Angeben von Bedingungen in einer Richtlinie. Die `aws:sourceIp` IPv4-Werte verwenden die CIDR-Standardnotation. Weitere Informationen finden Sie unter IP-Adressen-Bedingungsoperatoren im IAM-Benutzerhandbuch.

```
{
  "Version": "2012-10-17",
  "Id": "IBPolicyId1",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "imagebuilder.GetComponent:*",
      "Resource": "arn:aws:imagebuilder:::examplecomponent/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"},
         "NotIpAddress": {"aws:SourceIp": "54.240.143.188/32"}
      }
    }
  ]
}
```

Verwenden von verwalteten Richtlinien für EC2 Image Builder

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. AWS aktualisiert 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.

AWSImageBuilderFullAccess-Richtlinie

Die AWSImageBuilderFullAccess Richtlinie gewährt vollen Zugriff auf Image-Builder-Ressourcen für die Rolle, der sie zugeordnet ist, sodass die Rolle Image-Builder-Ressourcen auflisten, beschreiben, erstellen, aktualisieren und löschen kann. Die Richtlinie gewährt auch gezielte Berechtigungen für verwandte , AWS-Services die z. B. zur Überprüfung von Ressourcen oder zur Anzeige aktueller Ressourcen für das Konto in der benötigt werdenAWS Management Console.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Image Builder – Administrativer Zugriff wird gewährt, sodass die Rolle Image-Builder-Ressourcen auflisten, beschreiben, erstellen, aktualisieren und löschen kann.
- Amazon EC2 – Der Zugriff wird für Amazon EC2 Describe-Aktionen gewährt, die erforderlich sind, um das Vorhandensein von Ressourcen zu überprüfen oder Listen von Ressourcen abzurufen, die zum Konto gehören.
- IAM – Zugriff zum Abrufen und Verwenden von Instance-Profilen, deren Name „imagebuilder“ enthält, zum Überprüfen des Vorhandenseins der serviceverknüpften Rolle von Image Builder über die `iam:GetRole` API-Aktion und zum Erstellen der serviceverknüpften Rolle von Image Builder.
- License Manager – Zugriff zum Auflisten von Lizenzkonfigurationen oder Lizenzen für eine Ressource wird gewährt.
- Amazon S3 – Zugriff auf das Auflisten von Buckets, die zum Konto gehören, und Image-Builder-Buckets mit „imagebuilder“ in ihren Namen.

- Amazon SNS – Amazon SNS werden Schreibberechtigungen erteilt, um die Themeneigentümerschaft für Themen zu überprüfen, die „imagebuilder“ enthalten.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die -AWSImageBuilderFullAccessRichtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:ListTopics"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:*:*:*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "license-manager:ListLicenseConfigurations",
                "license-manager:ListLicenseSpecificationsForResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole"
```

```
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::*:instance-profile/*imagebuilder*",
                "arn:aws:iam::*:role/*imagebuilder*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
```

```
            ],
            "Resource": "arn:aws:s3::*:*imagebuilder*"
        },
        {
            "Action": "iam:CreateServiceLinkedRole",
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "imagebuilder.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeSnapshots",
                "ec2:DescribeVpcs",
                "ec2:DescribeRegions",
                "ec2:DescribeVolumes",
                "ec2:DescribeSubnets",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeLaunchTemplates"
            ],
            "Resource": "*"
        }
    ]
}
```

AWSImageBuilderReadOnlyAccess-Richtlinie

Die AWSImageBuilderReadOnlyAccess Richtlinie bietet schreibgeschützten Zugriff auf alle Image-Builder-Ressourcen. Berechtigungen zum Überprüfen, ob die serviceverknüpfte Rolle von Image Builder über die -`iam:GetRole`API-Aktion vorhanden ist, werden erteilt.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Image Builder – Der Zugriff auf Image-Builder-Ressourcen wird für den schreibgeschützten Zugriff gewährt.
- IAM – Der Zugriff zur Überprüfung des Vorhandenseins der serviceverknüpften Rolle von Image Builder über die -`iam:GetRole`API-Aktion wird gewährt.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die -AWSImageBuilderReadOnlyAccessRichtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:Get*",
                "imagebuilder:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
        }
    ]
}
```

AWSServiceRoleForImageBuilder-Richtlinie

Die AWSServiceRoleForImageBuilder Richtlinie ermöglicht es Image Builder, AWS-Services in Ihrem Namen aufzurufen.

Details zu Berechtigungen

Diese Richtlinie wird an die serviceverknüpfte Rolle von Image Builder angehängt, wenn die Rolle über Systems Manager erstellt wird. Informationen zum Überprüfen bestimmter Berechtigungen, die gewährt werden, finden Sie im Richtlinienbeispiel in diesem Abschnitt. Weitere Informationen

zur serviceverknüpften Rolle von Image Builder finden Sie unter Verwenden von serviceverknüpften Rollen für EC2 Image Builder.

Die Richtlinie umfasst die folgenden Berechtigungen:

- CloudWatch Protokolle – Zugriff zum Erstellen und Hochladen von CloudWatch Protokollen in jede Protokollgruppe, deren Name mit beginnt`/aws/imagebuilder/`.
- Amazon EC2 – Image Builder erhält Zugriff, um Images zu erstellen und EC2-Instances in Ihrem Konto zu starten, wobei zugehörige Snapshots, Volumes, Netzwerkschnittstellen, Subnetze, Sicherheitsgruppen, Lizenzkonfiguration und Schlüsselpaare verwendet werden, sofern das Image, die Instance und die Volumes, die erstellt oder verwendet werden, mit `CreatedBy: EC2 Image Builder` oder gekennzeichnet sind`CreatedBy: EC2 Fast Launch`.

  Image Builder kann Informationen über Amazon EC2-Images, Instance-Attribute, Instance-Status, die Instance-Typen, die für Ihr Konto verfügbar sind, Startvorlagen, Subnetze, Hosts und Tags auf Ihren Amazon EC2Ressourcen abrufen.

  Image Builder kann Image-Einstellungen aktualisieren, um das schnellere Starten von Windows-Instances in Ihrem Konto zu aktivieren oder zu deaktivieren, wobei das Image mit gekennzeichnet ist`CreatedBy: EC2 Image Builder`.

  Darüber hinaus kann Image Builder Instances starten, stoppen und beenden, die in Ihrem Konto ausgeführt werden, Amazon-EBS-Snapshots freigeben, Images erstellen und aktualisieren und Vorlagen starten, vorhandene Images abmelden, Tags hinzufügen und Images kontenübergreifend replizieren, denen Sie über die Ec2ImageBuilderCrossAccountDistributionAccess Richtlinie Berechtigungen erteilt haben. Image Builder-Markierung ist für alle diese Aktionen erforderlich, wie zuvor beschrieben.
- Amazon ECR – Image Builder erhält Zugriff, um bei Bedarf ein Repository für Container-Image-Schwachstellenscans zu erstellen und die Ressourcen zu markieren, die es erstellt, um den Umfang seiner Vorgänge einzuschränken. Image Builder erhält auch Zugriff, um die Container-Images zu löschen, die es für die Scans erstellt hat, nachdem es Snapshots der Schwachstellen erstellt hat.
- EventBridge – Image Builder wird Zugriff zum Erstellen und Verwalten von EventBridge Regeln gewährt.
- IAM – Image Builder erhält Zugriff, um jede Rolle in Ihrem Konto an Amazon EC2 und an VM Import/Export zu übergeben.

- Amazon Inspector – Image Builder erhält Zugriff, um zu bestimmen, wann Amazon Inspector Build-Instance-Scans abschließt, und um Ergebnisse für Images zu sammeln, die so konfiguriert sind, dass sie dies zulassen.
- AWS KMS – Amazon EBS wird Zugriff zum Verschlüsseln, Entschlüsseln oder erneuten Verschlüsseln von Amazon-EBS-Volumes gewährt. Dies ist wichtig, um sicherzustellen, dass verschlüsselte Volumes funktionieren, wenn Image Builder ein Image erstellt.
- License Manager – Image Builder erhält Zugriff auf die Aktualisierung der License Manager-Spezifikationen über `license-manager:UpdateLicenseSpecificationsForResource`.
- Amazon SNS – Schreibberechtigungen werden für jedes Amazon SNS-Thema in Ihrem Konto erteilt.
- Systems Manager – Image Builder erhält Zugriff, um Systems Manager-Befehle und deren Aufrufe, Bestandseinträge , Instance-Informationen und Automatisierungsausführungsstatus zu beschreiben und Details zum Befehlsaufruf abzurufen. Image Builder kann auch Automatisierungssignale senden und Automatisierungsausführungen für jede Ressource in Ihrem Konto stoppen.

  Image Builder kann Run-Befehlsaufrufe an jede Instance ausgeben`AWS-RunPowerShellScript`, die `"CreatedBy": "EC2 Image Builder"` für die folgenden Skriptdateien markiert ist: `AWS-RunShellScript`, oder `AWSEC2-RunSysprep`. Image Builder kann eine Systems Manager-Automatisierungsausführung in Ihrem Konto für Automatisierungsdokumente starten, bei denen der Name mit beginnt`ImageBuilder`.

  Image Builder ist auch in der Lage, State Manager-Zuordnungen für jede Instance in Ihrem Konto zu erstellen oder zu löschen, solange das Zuordnungsdokument ist`AWS-GatherSoftwareInventory`, und die Systems Manager serviceverknüpfte Rolle in Ihrem Konto zu erstellen.
- AWS STS – Image Builder erhält Zugriff, um Rollen mit dem Namen EC2ImageBuilderDistributionCrossAccountRole von Ihrem Konto für jedes Konto zu übernehmen, für das die Vertrauensrichtlinie für die Rolle dies zulässt. Dies wird für die kontoübergreifende Image-Verteilung verwendet.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die -AWSServiceRoleForImageBuilderRichtlinie.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:launch-template/*",
                "arn:aws:license-manager:*:*:license-configuration:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
```

```
                    "vmie.amazonaws.com"
                ]
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:StopInstances",
            "ec2:StartInstances",
            "ec2:TerminateInstances"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:CopyImage",
            "ec2:CreateImage",
            "ec2:CreateLaunchTemplate",
            "ec2:DeregisterImage",
            "ec2:DescribeImages",
            "ec2:DescribeInstanceAttribute",
            "ec2:DescribeInstanceStatus",
            "ec2:DescribeInstances",
            "ec2:DescribeInstanceTypeOfferings",
            "ec2:DescribeInstanceTypes",
            "ec2:DescribeSubnets",
            "ec2:DescribeTags",
            "ec2:ModifyImageAttribute",
            "ec2:DescribeImportImageTasks",
            "ec2:DescribeExportImageTasks",
            "ec2:DescribeSnapshots",
            "ec2:DescribeHosts"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
```

```
            "Action": [
                "ec2:ModifySnapshotAttribute"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "RunInstances",
                        "CreateImage"
                    ],
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*:*:export-image-task/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
```

```
            "Resource": [
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:launch-template/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "license-manager:UpdateLicenseSpecificationsForResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:ListCommands",
                "ssm:ListCommandInvocations",
                "ssm:AddTagsToResource",
                "ssm:DescribeInstanceInformation",
                "ssm:GetAutomationExecution",
                "ssm:StopAutomationExecution",
                "ssm:ListInventoryEntries",
                "ssm:SendAutomationSignal",
                "ssm:DescribeInstanceAssociationsStatus",
                "ssm:DescribeAssociationExecutions",
                "ssm:GetCommandInvocation"
            ],
            "Resource": "*"
        },
```

```
        {
            "Effect": "Allow",
            "Action": "ssm:SendCommand",
            "Resource": [
                "arn:aws:ssm:*:*:document/AWS-RunPowerShellScript",
                "arn:aws:ssm:*:*:document/AWS-RunShellScript",
                "arn:aws:ssm:*:*:document/AWSEC2-RunSysprep",
                "arn:aws:s3:::*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:resourceTag/CreatedBy": [
                        "EC2 Image Builder"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": "arn:aws:ssm:*:*:automation-definition/ImageBuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:CreateAssociation",
                "ssm:DeleteAssociation"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/AWS-GatherSoftwareInventory",
                "arn:aws:ssm:*:*:association/*",
                "arn:aws:ec2:*:*:instance/*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo",
                "kms:GenerateDataKeyWithoutPlaintext"
            ],
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "kms:EncryptionContextKeys": [
                        "aws:ebs:id"
                    ]
                },
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "kms:CreateGrant",
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": true
                },
```

```
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::*:role/
EC2ImageBuilderDistributionCrossAccountRole"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplateVersion",
                "ec2:DescribeLaunchTemplates",
                "ec2:ModifyLaunchTemplate",
                "ec2:DescribeLaunchTemplateVersions"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ExportImage"
            ],
            "Resource": "arn:aws:ec2:*::image/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ExportImage"
            ],
            "Resource": "arn:aws:ec2:*:*:export-image-task/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CancelExportTask"
            ],
            "Resource": "arn:aws:ec2:*:*:export-image-task/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "ssm.amazonaws.com",
                        "ec2fastlaunch.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:EnableFastLaunch"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*:*:launch-template/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "inspector2:ListCoverage",
                "inspector2:ListFindings"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:CreateRepository"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:TagResource"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/image-builder-*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchDeleteImage"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/image-builder-*",
            "Condition": {
                "StringEquals": {
                    "ecr:ResourceTag/CreatedBy": "EC2 Image Builder"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "events:DeleteRule",
                "events:DescribeRule",
                "events:PutRule",
                "events:PutTargets",
                "events:RemoveTargets"
            ],
            "Resource": [
                "arn:aws:events:*:*:rule/ImageBuilder-*"
            ]
        }
    ]
}
```

Ec2ImageBuilderCrossAccountDistributionAccess-Richtlinie

Die Ec2ImageBuilderCrossAccountDistributionAccess Richtlinie gewährt Image Builder Berechtigungen zum Verteilen von Images auf Konten in Zielregionen. Darüber hinaus kann Image Builder Tags beschreiben, kopieren und auf jedes Amazon EC2-Image im Konto anwenden. Die Richtlinie gewährt auch die Möglichkeit, AMI-Berechtigungen über die API`ec2:ModifyImageAttribute`-Aktion zu ändern.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Amazon EC2 – Amazon EC2 erhält Zugriff zum Beschreiben, Kopieren und Ändern von Attributen für ein Image und zum Erstellen von Tags für Amazon EC2-Images im Konto.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die - Ec2ImageBuilderCrossAccountDistributionAccessRichtlinie.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*::image/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:CopyImage",
                "ec2:ModifyImageAttribute"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2ImageBuilderLifecycleExecutionPolicy-Richtlinie

Die EC2ImageBuilderLifecycleExecutionPolicy Richtlinie gewährt Image Builder Berechtigungen zum Ausführen von Aktionen wie Veralten, Deaktivieren oder Löschen von Image-Builder-Image-Ressourcen und ihren zugrunde liegenden Ressourcen (AMIs, Snapshots), um automatisierte Regeln für Image-Lebenszyklusverwaltungsaufgaben zu unterstützen.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Amazon EC2 – Amazon EC2 erhält Zugriff, um die folgenden Aktionen für Amazon Machine Images (AMIs) in dem Konto auszuführen, das mit gekennzeichnet ist`CreatedBy: EC2 Image Builder`.
  - Aktivieren und deaktivieren Sie ein AMI.
  - Aktivieren und deaktivieren Sie die Image-Veraltung.
  - Beschreiben Sie ein AMI und heben Sie die Registrierung auf.
  - Beschreiben und ändern Sie AMI-Image-Attribute.
  - Löschen Sie Volume-Snapshots, die dem AMI zugeordnet sind.
  - Abrufen von Tags für eine Ressource.
  - Hinzufügen oder Entfernen von Tags aus einem AMI zur Veralterung.

- Amazon ECR – Amazon ECR wird Zugriff gewährt, um die folgenden Batch-Aktionen auf ECR-Repositorys mit dem -`LifecycleExecutionAccess: EC2 Image Builder`Tag durchzuführen. Batch-Aktionen unterstützen automatisierte Container-Image-Lebenszyklusregeln.
  - `ecr:BatchGetImage`
  - `ecr:BatchDeleteImage`

  Der Zugriff für ECR-Repositorys, die mit gekennzeichnet sind, wird auf Repository-Ebene gewährt`LifecycleExecutionAccess: EC2 Image Builder`.
- AWS Ressourcengruppen – Image Builder wird Zugriff gewährt, um Ressourcen basierend auf Tags abzurufen.
- EC2 Image Builder – Image Builder wird Zugriff zum Löschen von Image-Builder-Image-Ressourcen gewährt.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die -EC2ImageBuilderLifecycleExecutionPolicyRichtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Ec2ImagePermission",
            "Effect": "Allow",
            "Action": [
                "ec2:EnableImage",
                "ec2:DeregisterImage",
                "ec2:EnableImageDeprecation",
                "ec2:DescribeImageAttribute",
                "ec2:DisableImage",
                "ec2:DisableImageDeprecation"
            ],
            "Resource": "arn:aws:ec2:*::image/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "EC2DeleteSnapshotPermission",
```

```
            "Effect": "Allow",
            "Action": "ec2:DeleteSnapshot",
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "EC2TagsPermission",
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteTags",
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*::image/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/DeprecatedBy": "EC2 Image Builder",
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "DeprecatedBy"
                }
            }
        },
        {
            "Sid": "ECRImagePermission",
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:BatchDeleteImage"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/*",
            "Condition": {
                "StringEquals": {
                    "ecr:ResourceTag/LifecycleExecutionAccess": "EC2 Image Builder"
                }
            }
        },
```

```
        {
            "Sid": "ImageBuilderEC2TagServicePermission",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "tag:GetResources",
                "imagebuilder:DeleteImage"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2InstanceProfileForImageBuilder-Richtlinie

Die EC2InstanceProfileForImageBuilder Richtlinie gewährt die Mindestberechtigungen, die eine EC2-Instance für die Arbeit mit Image Builder benötigt. Dies schließt keine Berechtigungen ein, die für die Verwendung des Systems Manager Agent erforderlich sind.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- CloudWatch Protokolle – Zugriff zum Erstellen und Hochladen von CloudWatch Protokollen in jede Protokollgruppe, deren Name mit beginnt`/aws/imagebuilder/`.
- Image Builder – Zugriff zum Abrufen einer beliebigen Image-Builder-Komponente wird gewährt.
- AWS KMS – Zugriff zum Entschlüsseln einer Image-Builder-Komponente wird gewährt, wenn sie mit verschlüsselt wurdeAWS KMS.
- Amazon S3 – Der Zugriff auf Objekte wird gewährt, die in einem Amazon S3-Bucket gespeichert werden, dessen Name mit beginnt`ec2imagebuilder-`.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die -EC2InstanceProfileForImageBuilderRichtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "imagebuilder:GetComponent"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "kms:EncryptionContextKeys": "aws:imagebuilder:arn",
                    "aws:CalledVia": [
                        "imagebuilder.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::ec2imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        }
    ]
}
```

EC2InstanceProfileForImageBuilderECRContainerBuilds-Richtlinie

Die EC2InstanceProfileForImageBuilderECRContainerBuilds Richtlinie gewährt die Mindestberechtigungen, die für eine EC2-Instance erforderlich sind, wenn Sie mit Image Builder arbeiten, um Docker-Images zu erstellen und die Images dann in einem Amazon-ECR-Container-

Repository zu registrieren und zu speichern. Dies schließt keine Berechtigungen ein, die für die Verwendung des Systems Manager Agent erforderlich sind.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- CloudWatch Protokolle – Zugriff zum Erstellen und Hochladen von CloudWatch Protokollen in jede Protokollgruppe, deren Name mit beginnt`/aws/imagebuilder/`.
- Amazon ECR – Amazon ECR wird Zugriff gewährt, um ein Container-Image abzurufen, zu registrieren und zu speichern und ein Autorisierungstoken abzurufen.
- Image Builder – Zugriff zum Abrufen einer Image-Builder-Komponente oder eines Container-Rezepts wird gewährt.
- AWS KMS – Der Zugriff zum Entschlüsseln einer Image-Builder-Komponente oder eines Container-Rezepts wird gewährt, wenn sie über verschlüsselt wurdeAWS KMS.
- Amazon S3 – Der Zugriff auf Objekte, die in einem Amazon S3-Bucket gespeichert sind, dessen Name mit beginnt, wird gewährt`ec2imagebuilder-`.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die -EC2InstanceProfileForImageBuilderECRContainerBuildsRichtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:GetComponent",
                "imagebuilder:GetContainerRecipe",
                "ecr:GetAuthorizationToken",
                "ecr:BatchGetImage",
                "ecr:InitiateLayerUpload",
                "ecr:UploadLayerPart",
                "ecr:CompleteLayerUpload",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:PutImage"
            ],
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "kms:EncryptionContextKeys": "aws:imagebuilder:arn",
                    "aws:CalledVia": [
                        "imagebuilder.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::ec2imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        }
    ]
}
```

Image Builder-Aktualisierungen für -AWSverwaltete Richtlinien

Dieser Abschnitt enthält Informationen zu Aktualisierungen für -AWSverwaltete Richtlinien für Image Builder, seit dieser Service mit der Verfolgung dieser Änderungen begonnen hat. Um automatische Warnungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der Dokumentverlaufsseite von Image Builder.

Änderung	Beschreibung	Datum
EC2ImageBuilderLifecycleExecutionPolicy – Neue Richtlinie.	Image Builder hat die neue `EC2ImageBuilderLifecycleExecutionPolicy` Richtlinie hinzugefügt, die Berechtigungen für die Verwaltung des Image-Lebenszyklus enthält.	17. November 2023
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen, um macOS-Unterstützung bereitzustellen. • ec2:DescribeHosts enable Image Builder wurde hinzugefügt, um die hostId abzufragen und festzustellen, wann sie sich in einem gültigen Status befindet, um eine Instance zu starten. • ssm:GetCommandInvocation, API-Aktion zur Verbesserung der Methode hinzugefügt, die Image Builder verwendet, um Details zum Befehlsaufruf abzurufen.	28. August 2023
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen, damit Image-Builder-Workflows Schwachstellenerge	30. März 2023

Änderung	Beschreibung	Datum
	bnisse sowohl für AMI- als auch für ECR-Container-Imag e-Builds sammeln können. Die neuen Berechtigungen unterstützen die CVE-Erken nungs- und Berichtsfunktion. • Inspector2:ListCoverage und Inspector2: ListFindings hinzugefügt, damit Image Builder bestimmen kann, wann Amazon Inspector Test-Instance-Scans abschließt, und Ergebnisse für Bilder sammeln kann, die so konfiguriert sind, dass sie dies zulassen. • ecr: wurde hinzugefü gtCreateRepository, wobei Image Builder das Repositor y mit `CreatedBy: EC2 Image Builder` () markieren musstag-on-create. Außerdem wurde ecr:TagResource (erforder lich für tag-on-create) mit derselben CreatedBy Tag-Einschränkung und einer zusätzlichen Einschrän kung hinzugefügt, für die der Repository-Name mit beginnen mussimage-bui lder-* . Die Namensein schränkung verhindert die	

Änderung	Beschreibung	Datum
	Eskalation von Rechten und verhindert Änderungen an Repositorys, die Image Builder nicht erstellt hat. • ecr:BatchDeleteImage for ECR-Repositorys hinzugefügt, die mit gekennzeichnet sind`CreatedBy: EC2 Image Builder`. Diese Berechtigung erfordert, dass der Repository-Name mit beginnt`image-builder-*`. • Ereignisberechtigungen für Image Builder hinzugefügt, um von Amazon EventBridge verwaltete Regeln zu erstellen und zu verwalten, die `ImageBuilder-*` im Namen enthalten.	
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen: • License Manager-Lizenzen als Ressource für den ec2:RunInstance call hinzugefügt, damit Kunden Basis-Image-AMIs verwenden können, die einer Lizenzkonfiguration zugeordnet sind.	22. März 2022

Änderung	Beschreibung	Datum
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen: • Berechtigungen für die EC2 EnableFastLaunch API-Aktion hinzugefügt, um schnelleres Starten für Windows-Instances zu aktivieren und zu deaktivieren. • Weitere Bereiche für ec2:CreateTags action- und Ressourcen-Tag-Bedingungen.	21. Februar 2022
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen: • Berechtigungen zum Aufrufen des VMIE-Service zum Importieren einer VM und zum Erstellen eines Basis-AMI daraus hinzugefügt. • Geltungsbereich für ec2:CreateTags action- und Ressourcen-Tag-Bedingungen.	20. November 2021

Änderung	Beschreibung	Datum
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat neue Berechtigungen hinzugefügt, um Probleme zu beheben, bei denen mehr als eine Bestandszuordnung dazu führt, dass der Image-Build hängen bleibt.	11. August 2021
AWSImageBuilderFullAccess – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Rolle für vollständigen Zugriff vorgenommen: • Berechtigungen zum Zulassen von hinzugefügtec2:DescribeInstanceTypeOffereings . • Dem Aufruf von wurden Berechtigungen hinzugefügtec2:DescribeInstanceTypeOffereings , damit die Image-Builder-Konsole die Instance-Typen, die im Konto verfügbar sind, genau wiedergibt.	13. April 2021
Image Builder hat mit der Verfolgung von Änderungen begonnen	Image Builder hat mit der Verfolgung von Änderungen für seine -AWSverwalteten Richtlinien begonnen.	02. April 2021

Verwenden von serviceverknüpften Rollen für EC2 Image Builder

EC2 Image Builder verwendet AWS Identity and Access Management (IAM) serviceverknüpfte Rollen . Eine serviceverknüpfte Rolle ist ein spezieller Typ einer IAM-Rolle, die direkt mit Image Builder verknüpft ist. Serviceverknüpfte Rollen werden von Image Builder vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle macht die Einrichtung von Image Builder effizienter, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Image Builder definiert die Berechtigungen seiner serviceverknüpften Rollen. Sofern keine andere Konfiguration festgelegt wurde, kann nur Image Builder die Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Die Berechtigungsrichtlinie kann an keine andere IAM-Entität angefügt werden.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter . AWS-Services Diese funktionieren mit IAM und suchen nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Image Builder

Image Builder verwendet die AWSServiceRoleForImageBuilder serviceverknüpfte Rolle , um EC2 Image Builder den Zugriff auf -AWSRessourcen in Ihrem Namen zu ermöglichen. Die serviceverknüpfte Rolle vertraut dem imagebuilder.amazonaws.com-Service, die Rolle zu übernehmen.

Sie müssen diese serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie Ihr erstes Image-Builder-Image in der -AWSManagementkonsoleAWS CLI, der oder der -AWSAPI erstellen, erstellt Image Builder die serviceverknüpfte Rolle für Sie.

Die folgenden Aktionen erstellen ein neues Image:

- Führen Sie den Pipeline-Assistenten in der Image-Builder-Konsole aus, um ein benutzerdefiniertes Image zu erstellen.
- Verwenden Sie eine der folgenden API-Aktionen oder den entsprechenden AWS CLI Befehl:
  - Die CreateImage API-Aktion (create-image in der AWS CLI).
  - Die ImportVmImage API-Aktion (import-vm-image in der AWS CLI).
  - Die StartImagePipelineExecution API-Aktion (start-image-pipeline-execution in der AWS CLI).

> **Important**
>
> Wenn die serviceverknüpfte Rolle aus Ihrem Konto gelöscht wird, können Sie sie mit demselben Verfahren erneut erstellen. Wenn Sie Ihre erste EC2 Image Builder-Ressource erstellen, erstellt Image Builder die serviceverknüpfte Rolle erneut für Sie.

Die Berechtigungen für die finden AWSServiceRoleForImageBuilderSie auf der Seite AWSServiceRoleForImageBuilder-Richtlinie. Weitere Informationen zum Konfigurieren von Berechtigungen für eine serviceverknüpfte Rolle finden Sie unter Serviceverknüpfte Rollenberechtigungen im IAM-Benutzerhandbuch.

Entfernen einer serviceverknüpften Image-Builder-Rolle aus Ihrem Konto

Sie können die IAM-Konsole, die oder die -AWSAPI verwendenAWS CLI, um die serviceverknüpfte Rolle für Image Builder manuell aus Ihrem Konto zu entfernen. Bevor Sie dies tun, müssen Sie jedoch sicherstellen, dass keine Image-Builder-Ressourcen aktiviert sind, die darauf verweisen.

> **Note**
>
> Wenn der Image-Builder-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.

Bereinigen von Image Builder-Ressourcen, die von der **`AWSServiceRoleForImageBuilder`** Rolle verwendet werden

1. Stellen Sie sicher, dass keine Pipeline-Builds ausgeführt werden, bevor Sie beginnen. Um einen laufenden Build abzubrechen, verwenden Sie den `cancel-image-creation` Befehl aus der AWS CLI.

   ```
   aws imagebuilder cancel-image-creation --image-build-version-
   arn arn:aws:imagebuilder:us-east-1:123456789012:image-pipeline/sample-pipeline
   ```

2. Ändern Sie alle Pipeline-Zeitpläne, um einen manuellen Build-Prozess zu verwenden, oder löschen Sie sie, wenn Sie sie nicht erneut verwenden werden. Weitere Informationen zum Löschen von Ressourcen finden Sie unter Löschen von EC2 Image Builder-Ressourcen.

Löschen der serviceverknüpften Rolle mit IAM

Sie können die IAM-Konsole, die oder die AWS-API verwendenAWS CLI, um die `AWSServiceRoleForImageBuilder` Rolle aus Ihrem Konto zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte EC2 Image Builder-Rollen

Image Builder unterstützt die Verwendung von serviceverknüpften Rollen in allen AWS Regionen, in denen der Service verfügbar ist. Eine Liste der unterstützten AWS Regionen finden Sie unter Regionen und Endpunkte von AWS.

Fehlerbehebung für EC2 Image Builder-Identität und -Zugriff

Themen

- Ich bin nicht autorisiert, eine Aktion in Image Builder auszuführen
- Ich bin nicht autorisiert, iam durchzuführen:PassRole
- Ich möchte Personen außerhalb meines AWS-Konto Zugriff auf meine Image-Builder-Ressourcen gewähren

Ich bin nicht autorisiert, eine Aktion in Image Builder auszuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `imagebuilder:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 imagebuilder:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `imagebuilder:`*`GetWidget`*-Aktion auf die *`my-example-widget`*-Ressource zugreifen kann.

Wenden Sie sich an Ihren AWS-Administrator, falls Sie weitere Unterstützung benötigen. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht autorisiert, iam durchzuführen:PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zum Ausführen der `iam:PassRole` Aktion autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, um eine Rolle an Image Builder übergeben zu können.

Einige AWS-Services erlauben die Übergabe einer vorhandenen Rolle an diesen Dienst, sodass keine neue Servicerolle oder serviceverknüpfte Rolle erstellt werden muss. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Image Builder auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenden Sie sich an Ihren AWS-Administrator, falls Sie weitere Unterstützung benötigen. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb meines AWS-Konto Zugriff auf meine Image-Builder-Ressourcen gewähren

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen dazu, ob Image Builder diese Funktionen unterstützt, finden Sie unter Funktionsweise von EC2 Image Builder mit IAM.
- Informationen zum Gewähren des Zugriffs auf Ihre Ressourcen für alle Ihre AWS-Konten finden Sie unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen Ihrer AWS-Konto im IAM-Benutzerhandbuch.

- Informationen dazu, wie Sie AWS-Konten-Drittanbieter Zugriff auf Ihre Ressourcen bereitstellen, finden Sie unter Gewähren des Zugriffs auf AWS-Konten von externen Benutzern im IAM-Benutzerhandbuch.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.

Compliance-Validierung für EC2 Image Builder

EC2 Image Builder fällt nicht in den Geltungsbereich von AWS Compliance-Programmen.

Eine Liste der AWS-Services, die in den Anwendungsbereich bestimmter Compliance-Programme fallen, finden Sie unter AWS-Services im Anwendungsbereich nach Compliance-Programm . Allgemeine Informationen finden Sie unter AWS-Compliance-Programme.

Sie können Auditberichte von Drittanbietern unter AWS Artifact herunterladen. Weitere Informationen finden Sie unter Herunterladen von Berichten in AWS Artifact .

Ihre Compliance-Verantwortung bei der Verwendung von Image Builder hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS stellt die folgenden Ressourcen zur Unterstützung der Compliance bereit:

- Quick-Start-Anleitungen für Sicherheit und Compliance – In diesen Bereitstellungsleitfäden finden Sie wichtige Überlegungen zur Architektur sowie die einzelnen Schritte zur Bereitstellung von Sicherheits- und Compliance-orientierten Basisumgebungen in AWS.
- AWS-Compliance-Ressourcen – Diese Arbeitsbücher und Leitfäden könnten für Ihre Branche und Ihren Standort relevant sein.
- Auswertung von Ressourcen mit Regeln im AWS ConfigEntwicklerhandbuch – Der AWS Config-Service bewertet, wie gut Ihre Ressourcenkonfigurationen mit internen Praktiken, Branchenrichtlinien und Vorschriften übereinstimmen.
- AWS Security Hub – Dieser AWS-Dienst liefert einen umfassenden Überblick über den Sicherheitsstatus in AWS. So können Sie die Compliance mit den Sicherheitsstandards in der Branche und den bewährten Methoden abgleichen.

Sie können Compliance-Produkte von AWS Marketplace oder Komponenten von AWS Task Orchestrator and Executor (AWSTOE) in Ihre Image-Builder-Images integrieren, um sicherzustellen, dass Ihre Images konform sind. Weitere Informationen finden Sie unter Compliance-Produkte für Ihre Image-Builder-Images.

Ausfallsicherheit in EC2 Image Builder

Im Zentrum der globalen AWS Infrastruktur stehen die AWS-Regionen und Availability Zones (Verfügbarkeitszonen, AZs). AWS -Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die über hoch redundante Netzwerke mit niedriger Latenz und hohen Durchsätzen verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Mit dem EC2 Image Builder-Service können Sie Images, die in einer Region erstellt wurden, an andere Regionen verteilen, was ihnen eine multiregionale Ausfallsicherheit für AMIs bietet. Es gibt keinen Mechanismus zum „Backup“ von Image-Pipelines, Rezepten oder Komponenten. Sie können die Rezept- und Komponentendokumente außerhalb des Image-Builder-Services speichern, z. B. in einem Amazon S3-Bucket.

Der EC2 Image Builder kann nicht für Hochverfügbarkeit (HA) konfiguriert werden. Sie können Images an mehrere Regionen verteilen, um die Images hochverfügbarer zu machen.

Weitere Informationen über AWS-Regionen und -Availability Zones finden Sie unter AWS Globale Infrastruktur.

Infrastruktursicherheit in Image Builder

Das AWS globale Netzwerk bietet Sicherheitsfunktionen und steuert den Netzwerkzugriff für Services wie EC2 Image Builder. Weitere Informationen zur Infrastruktursicherheit, die für seine Services AWS bietet, finden Sie im Abschnitt Infrastruktursicherheit im Whitepaper Einführung in die AWS Sicherheit.

Um Anforderungen über das AWS globale Netzwerk für Image-Builder-API-Aktionen zu senden, muss Ihre Client-Software den folgenden Sicherheitsrichtlinien entsprechen:

- Um Anforderungen für Image-Builder-API-Aktionen zu senden, muss die Client-Software eine unterstützte Version von Transport Layer Security (TLS) verwenden.

> **Note**
>
> AWS stellt die Unterstützung für die TLS-Versionen 1.0 und 1.1 ein. Wir empfehlen dringend, Ihre Client-Software für die Verwendung von TLS Version 1.2 oder höher zu aktualisieren, damit Sie weiterhin eine Verbindung herstellen können. Weitere Informationen finden Sie in diesem AWS Security Blog-Beitrag.

- Client-Software muss Verschlüsselungssammlungen mit Perfect Forward Secrecy (PFS) unterstützen, z. B. kurzlebiges Diffie-Hellman (DHE) oder elliptische Kurven-Ephemeral Diffie-Hellman (ECDHE). Die meisten aktuellen Systeme, wie Java 7 und höher, unterstützen diese Modi.
- Sie müssen Ihre API-Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signieren, der einem AWS Identity and Access Management (IAM)-Prinzipal zugeordnet ist. Oder Sie können die AWS Security Token Service (AWS STS) verwenden, um temporäre Sicherheitsanmeldeinformationen für Ihre Anfragen zu generieren.

Darüber hinaus müssen die EC2-Instances, die Image Builder zum Erstellen und Testen von Images verwendet, Zugriff auf habenAWS Systems Manager.

Patch-Verwaltung in EC2 Image Builder

EC2 Image Builder bietet die neuesten AMIs von Amazon Linux 2, Amazon Linux 2023, Red Hat Enterprise Linux (RHEL), CentOS , Ubuntu, SUSE Linux Enterprise Server und Windows 2012 R2 und höher als verwaltete Image-Quellen. AMIs Sie behalten die Verantwortung für das Patchen des Amazon EC2-Systems gemäß dem Modell der geteilten Verantwortung bei. Wenn die EC2-Instances in Ihrem Anwendungs-Workload leicht ersetzt werden können, ist es möglicherweise effizienter, das Basis-AMI zu aktualisieren und alle Datenverarbeitungsknoten basierend auf diesem Image erneut bereitzustellen.

Im Folgenden finden Sie zwei Möglichkeiten, Ihre Image-Builder-AMIs auf dem neuesten Stand zu halten.

- AWSVon bereitgestellte Patching-Komponenten – EC2 Image Builder bietet zwei Build-Komponenten, `update-linux` und `update-windows`, die alle ausstehenden Betriebssystem-Updates installieren. Diese Komponenten verwenden das -`UpdateOS`Aktionsmodul. Weitere Informationen finden Sie unter UpdateOS. Die Komponenten können Ihren Image-BuildAWS-

Pipelines hinzugefügt werden, indem Sie sie aus der Liste der von bereitgestellten Komponenten auswählen.

- Benutzerdefinierte Build-Komponenten mit Patching-Operationen – Um Patches selektiv auf Betriebssystemen unterstützter AMIs zu installieren oder zu aktualisieren, können Sie eine Image-Builder-Komponente erstellen, um die erforderlichen Patches zu installieren. Eine benutzerdefinierte Komponente kann Patches mithilfe von Shell-Skripten (Bash oder PowerShell) installieren oder das `UpdateOS` Aktionsmodul verwenden, um Patches für die Installation oder den Ausschluss anzugeben. Weitere Informationen finden Sie unter Vom AWSTOE Komponentenmanager unterstützte Aktionsmodule.

  Komponente, die das -`UpdateOS`Aktionsmodul verwendet (Linux und Windows)

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: UpdateOS
  action: UpdateOS
```

  Komponente, die Bash zum Installieren von Yum-Updates verwendet

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: InstallYumUpdates
  action: ExecuteBash
  inputs:
    commands:
   - sudo yum update -y
```

Bewährte Methoden für die Sicherheit in EC2 Image Builder

EC2 Image Builder bietet eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden sind allgemeine Richtlinien und keine vollständige Sicherheitslösung. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Überlegungen und nicht als bindend ansehen.

- Verwenden Sie keine übermäßig zulässigen Sicherheitsgruppen in Image-Builder-Rezepten.
- Teilen Sie Images nicht mit Konten, denen Sie nicht vertrauen.
- Machen Sie Images, die private oder sensible Daten enthalten, nicht öffentlich.
- Wenden Sie alle verfügbaren Windows- oder Linux-Sicherheitspatches während Image-Builds an.

Wir empfehlen dringend, Ihre Images zu testen, um den Sicherheitsstatus und die geltenden Sicherheits-Compliance-Level zu überprüfen. Lösungen wie Amazon Inspector können dazu beitragen, den Sicherheits- und Compliance-Status von Images zu überprüfen.

Pipelines von IMDSv2 für Image Builder

Wenn Ihre Image-Builder-Pipeline ausgeführt wird, sendet sie HTTP-Anforderungen an EC2-Instances, die Image Builder zum Erstellen und Testen Ihres Images verwendet. Um die Version des IMDS zu konfigurieren, die Ihre Pipeline für die Startanforderungen verwendet, legen Sie den `httpTokens` Parameter in den Metadateneinstellungen Ihrer Image-Builder-Infrastrukturkonfigurations-Instance fest.

> **Note**
>
> Wir empfehlen, alle EC2-Instances, die Image Builder von einem Pipeline-Build aus startet, für die Verwendung von IMDSv2 zu konfigurieren, damit Abrufanforderungen für Instance-Metadaten einen signierten Token-Header erfordern.

Weitere Informationen zur Konfiguration der Image-Builder-Infrastruktur finden Sie unter Verwalten der Infrastrukturkonfiguration von EC2 Image Builder. Weitere Informationen zu EC2-Instance-Metadatenoptionen für Linux-Images finden Sie unter Konfigurieren der Instance-Metadatenoptionen im Amazon EC2-Benutzerhandbuch für Linux-Instances. Informationen zu Windows-Abbildern finden Sie unter Konfigurieren der Instance-Metadatenoptionen im Amazon EC2-Benutzerhandbuch für Windows-Instances.

Erforderliche Bereinigung nach der Erstellung

Nachdem Image Builder alle Build-Schritte für Ihr benutzerdefiniertes Image abgeschlossen hat, bereitet Image Builder die Build-Instance zum Testen und Erstellen von Images vor. Bevor Sie die Build-Instance herunterfahren, um den Snapshot zu erstellen, führt Image Builder die folgende Bereinigung durch, um die Sicherheit Ihres Images zu gewährleisten:

Linux

Die Image-Builder-Pipeline führt ein Bereinigungsskript aus, um sicherzustellen, dass das endgültige Image den bewährten Sicherheitsmethoden entspricht, und um alle Build-Artefakte oder -Einstellungen zu entfernen, die nicht auf Ihren Snapshot übertragen werden sollten. Sie können jedoch Abschnitte des Skripts überspringen oder die Benutzerdaten vollständig überschreiben. Daher entsprechen die von Image-Builder-Pipelines erstellten Images nicht unbedingt bestimmten regulatorischen Kriterien.

Wenn die Pipeline ihre Entwicklungs- und Testphasen abgeschlossen hat, führt Image Builder automatisch das folgende Bereinigungsskript aus, kurz bevor es das Ausgabe-Image erstellt.

> **Important**
>
> Wenn Sie Benutzerdaten in Ihrem Rezept überschreiben, wird das Skript nicht ausgeführt. Stellen Sie in diesem Fall sicher, dass Sie einen Befehl in Ihre Benutzerdaten aufnehmen, der eine leere Datei mit dem Namen erstellt`perform_cleanup`. Image Builder erkennt diese Datei und führt das Bereinigungsskript aus, bevor das neue Image erstellt wird.

```
#!/bin/bash
if [[ ! -f {{workingDirectory}}/perform_cleanup ]]; then
    echo "Skipping cleanup"
    exit 0
else
    sudo rm -f {{workingDirectory}}/perform_cleanup
fi

function cleanup() {
    FILES=("$@")
    for FILE in "${FILES[@]}"; do
        if [[ -f "$FILE" ]]; then
            echo "Deleting $FILE";
            sudo shred -zuf $FILE;
        fi;
        if [[ -f $FILE ]]; then
            echo "Failed to delete '$FILE'. Failing."
            exit 1
        fi;
    done
};
```

```
# Clean up for cloud-init files
CLOUD_INIT_FILES=(
    "/etc/sudoers.d/90-cloud-init-users"
    "/etc/locale.conf"
    "/var/log/cloud-init.log"
    "/var/log/cloud-init-output.log"
)
if [[ -f {{workingDirectory}}/skip_cleanup_cloudinit_files ]]; then
    echo "Skipping cleanup of cloud init files"
else
    echo "Cleaning up cloud init files"
    cleanup "${CLOUD_INIT_FILES[@]}"
    if [[ $( sudo find /var/lib/cloud -type f | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting files within /var/lib/cloud/*"
        sudo find /var/lib/cloud -type f -exec shred -zuf {} \;
    fi;

    if [[ $( sudo ls /var/lib/cloud | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/lib/cloud/*"
        sudo rm -rf /var/lib/cloud/* || true
    fi;
fi;


# Clean up for temporary instance files
INSTANCE_FILES=(
    "/etc/.updated"
    "/etc/aliases.db"
    "/etc/hostname"
    "/var/lib/misc/postfix.aliasesdb-stamp"
    "/var/lib/postfix/master.lock"
    "/var/spool/postfix/pid/master.pid"
    "/var/.updated"
    "/var/cache/yum/x86_64/2/.gpgkeyschecked.yum"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_files ]]; then
    echo "Skipping cleanup of instance files"
else
    echo "Cleaning up instance files"
    cleanup "${INSTANCE_FILES[@]}"
fi;
```

```
# Clean up for ssh files
SSH_FILES=(
    "/etc/ssh/ssh_host_rsa_key"
    "/etc/ssh/ssh_host_rsa_key.pub"
    "/etc/ssh/ssh_host_ecdsa_key"
    "/etc/ssh/ssh_host_ecdsa_key.pub"
    "/etc/ssh/ssh_host_ed25519_key"
    "/etc/ssh/ssh_host_ed25519_key.pub"
    "/root/.ssh/authorized_keys"
)
if [[ -f {{workingDirectory}}/skip_cleanup_ssh_files ]]; then
    echo "Skipping cleanup of ssh files"
else
    echo "Cleaning up ssh files"
    cleanup "${SSH_FILES[@]}"
    USERS=$(ls /home/)
    for user in $USERS; do
        echo Deleting /home/"$user"/.ssh/authorized_keys;
        sudo find /home/"$user"/.ssh/authorized_keys -type f -exec shred -zuf {} \;
    done
    for user in $USERS; do
        if [[ -f /home/"$user"/.ssh/authorized_keys ]]; then
            echo Failed to delete /home/"$user"/.ssh/authorized_keys;
            exit 1
        fi;
    done;
fi;


# Clean up for instance log files
INSTANCE_LOG_FILES=(
    "/var/log/audit/audit.log"
    "/var/log/boot.log"
    "/var/log/dmesg"
    "/var/log/cron"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_log_files ]]; then
    echo "Skipping cleanup of instance log files"
else
    echo "Cleaning up instance log files"
    cleanup "${INSTANCE_LOG_FILES[@]}"
fi;
```

```
# Clean up for TOE files
if [[ -f {{workingDirectory}}/skip_cleanup_toe_files ]]; then
    echo "Skipping cleanup of TOE files"
else
    echo "Cleaning TOE files"
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type f | sudo wc -l) -gt 0 ]];
 then
        echo "Deleting files within {{workingDirectory}}/TOE_*"
        sudo find {{workingDirectory}}/TOE_* -type f -exec shred -zuf {} \;
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type f | sudo wc -l) -gt 0 ]];
 then
        echo "Failed to delete {{workingDirectory}}/TOE_*"
        exit 1
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type d | sudo wc -l) -gt 0 ]];
 then
        echo "Deleting {{workingDirectory}}/TOE_*"
        sudo rm -rf {{workingDirectory}}/TOE_*
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type d | sudo wc -l) -gt 0 ]];
 then
        echo "Failed to delete {{workingDirectory}}/TOE_*"
        exit 1
    fi
fi

# Clean up for ssm log files
if [[ -f {{workingDirectory}}/skip_cleanup_ssm_log_files ]]; then
    echo "Skipping cleanup of ssm log files"
else
    echo "Cleaning up ssm log files"
    if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
        echo "Deleting files within /var/log/amazon/ssm/*"
        sudo find /var/log/amazon/ssm -type f -exec shred -zuf {} \;
    fi
    if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
        echo "Failed to delete /var/log/amazon/ssm"
        exit 1
    fi
    if [[ -d "/var/log/amazon/ssm" ]]; then
        echo "Deleting /var/log/amazon/ssm/*"
        sudo rm -rf /var/log/amazon/ssm
    fi
```

```
    if [[ -d "/var/log/amazon/ssm" ]]; then
        echo "Failed to delete /var/log/amazon/ssm"
        exit 1
    fi
fi


if [[ $( sudo find /var/log/sa/sa* -type f | sudo wc -l ) -gt 0 ]]; then
    echo "Deleting /var/log/sa/sa*"
    sudo shred -zuf /var/log/sa/sa*
fi
if [[ $( sudo find /var/log/sa/sa* -type f | sudo wc -l ) -gt 0 ]]; then
    echo "Failed to delete /var/log/sa/sa*"
    exit 1
fi

if [[ $( sudo find /var/lib/dhclient/dhclient*.lease -type f | sudo wc -l ) -gt
 0 ]]; then
        echo "Deleting /var/lib/dhclient/dhclient*.lease"
        sudo shred -zuf /var/lib/dhclient/dhclient*.lease
fi
if [[ $( sudo find /var/lib/dhclient/dhclient*.lease -type f | sudo wc -l ) -gt
 0 ]]; then
        echo "Failed to delete /var/lib/dhclient/dhclient*.lease"
        exit 1
fi

if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
        echo "Deleting files within /var/tmp/*"
        sudo find /var/tmp -type f -exec shred -zuf {} \;
fi
if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
        echo "Failed to delete /var/tmp"
        exit 1
fi
if [[ $( sudo ls /var/tmp | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/tmp/*"
        sudo rm -rf /var/tmp/*
fi

# Shredding is not guaranteed to work well on rolling logs

if [[ -f "/var/lib/rsyslog/imjournal.state" ]]; then
        echo "Deleting /var/lib/rsyslog/imjournal.state"
```

```
        sudo shred -zuf /var/lib/rsyslog/imjournal.state
        sudo rm -f /var/lib/rsyslog/imjournal.state
fi

if [[ $( sudo ls /var/log/journal/ | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/log/journal/*"
        sudo find /var/log/journal/ -type f -exec shred -zuf {} \;
        sudo rm -rf /var/log/journal/*
fi

sudo touch /etc/machine-id
```

Windows

Nachdem die Image Builder-Pipeline Windows-Images angepasst hat, führt sie das Microsoft Sysprep-Dienstprogramm aus. Diese Aktionen folgen AWS bewährten Methoden zum Verstärken und Bereinigen des Images .

Überschreiben des Linux-Bereinigungsskripts

Image Builder erstellt Images, die standardmäßig sicher sind und unseren bewährten Sicherheitsmethoden entsprechen. Einige fortgeschrittenere Anwendungsfälle erfordern jedoch möglicherweise, dass Sie einen oder mehrere Abschnitte des integrierten Bereinigungsskripts überspringen. Wenn Sie einen Teil der Bereinigung überspringen müssen, empfehlen wir dringend, Ihr Ausgabe-AMI zu testen, um die Sicherheit Ihres Images zu gewährleisten.

Important

Das Überspringen von Abschnitten im Bereinigungsskript kann dazu führen, dass vertrauliche Informationen, wie z. B. Details zum Besitzerkonto oder SSH-Schlüssel, im endgültigen Image und in jeder Instance, die aus diesem Image gestartet wird, enthalten sind. Es können auch Probleme beim Starten von in verschiedenen Availability Zones, Regionen oder Konten auftreten.

In der folgenden Tabelle werden die Abschnitte des Bereinigungsskripts, die in diesem Abschnitt gelöschten Dateien und die Dateinamen beschrieben, mit denen Sie einen Abschnitt kennzeichnen können, den Image Builder überspringen soll. Um einen bestimmten Abschnitt des Bereinigungsskripts zu überspringen, können Sie das CreateFile Komponentenaktionsmodul oder

einen -Befehl in Ihren Benutzerdaten (wenn überschrieben) verwenden, um eine leere Datei mit dem Namen zu erstellen, der in der Spalte Überspringen des Abschnittsdateinamens angegeben ist.

> **Note**
>
> Die Dateien, die Sie erstellen, um einen Abschnitt des Bereinigungsskripts zu überspringen, sollten keine Dateierweiterung enthalten. Wenn Sie beispielsweise den `CLOUD_INIT_FILES` Abschnitt des Skripts überspringen möchten, aber eine Datei mit dem Namen erstellen`skip_cleanup_cloudinit_files.txt`, erkennt Image Builder die überspringende Datei nicht.

Eingabe

Abschnitt „Bereinigen“	Dateien entfernt	Überspringen des Abschnitts Dateinamens

`CLOUD_INIT_FILES`	`/etc/sudoers.d/90-cloud-init-users`
	`/etc/locale.conf`
	`/var/log/cloud-init.log`
	`/var/log/cloud-init-output.log`
		`skip_cleanup_cloudinit_files`

`INSTANCE_FILES`	`/etc/.updated`
	`/etc/aliases.db`
	`/etc/hostname`
	`/var/lib/misc/postfix.aliasesdb-stamp`
	`/var/lib/postfix/master.lock`
		`skip_cleanup_instance_files`

Abschnitt „Bereinigen“	Dateien entfernt	Überspringen des Abschnitts Dateinamens

	`/var/spool/postfix/pid/master.pid`
	`/var/.updated`
	`/var/cache/yum/x86_64/2/.gpgkeyschecked.yum`	

`SSH_FILES`	`/etc/ssh/ssh_host_rsa_key`
	`/etc/ssh/ssh_host_rsa_key.pub`
	`/etc/ssh/ssh_host_ecdsa_key`
	`/etc/ssh/ssh_host_ecdsa_key.pub`
	`/etc/ssh/ssh_host_ed25519_key`
	`/etc/ssh/ssh_host_ed25519_key.pub`
	`/root/.ssh/authorized_keys`
	`/home/<all users>/.ssh/authorized_keys;`
		`skip_cleanup_ssh_files`

Abschnitt „Bereinigen“	Dateien entfernt	Überspringen des Abschnitts Dateinamens
INSTANCE_LOG_FILES	/var/log/audit/audit.log /var/log/boot.log /var/log/dmesg /var/log/cron	skip_cleanup_instance_log_files
TOE_FILES	{{workingDirectory}}/TOE_*	skip_cleanup_toe_files
SSM_LOG_FILES	/var/log/amazon/ssm/*	skip_cleanup_ssm_log_files

Fehlerbehebung bei EC2 Image Builder

EC2 Image Builder lässt sich AWS-Services zur Überwachung und Fehlerbehebung in integrieren, um Sie bei der Behebung von Image-Build-Problemen zu unterstützen. Image Builder verfolgt und zeigt den Fortschritt für jeden Schritt im Image-Erstellungsprozess an. Darüber hinaus kann Image Builder Protokolle an einen von Ihnen bereitgestellten Amazon S3-Speicherort exportieren.

Zur erweiterten Fehlerbehebung können Sie vordefinierte Befehle und Skripts mit AWS Systems Manager Run Command ausführen.

Inhalt

Fehlerbehebung bei Pipeline-Builds

Wenn ein Image-Builder-Pipeline-Build fehlschlägt, gibt Image Builder eine Fehlermeldung zurück, die den Fehler beschreibt. Image Builder gibt auch einen `workflow execution ID` in der Fehlermeldung zurück, z. B. den in der folgenden Beispielausgabe:

```
Workflow Execution ID: wf-12345abc-6789-0123-abc4-567890123abc failed with reason: …
```

Image Builder ordnet Image-Build-Aktionen durch eine Reihe von Schritten an, die für die Laufzeitphasen im Rahmen seines Standard-Image-Erstellungsprozesses definiert sind, und leitet sie weiter. Die Build- und Testphasen des Prozesses haben jeweils einen zugehörigen Workflow. Wenn Image Builder einen Workflow ausführt, um ein neues Image zu erstellen oder zu testen, wird eine Workflow-Metadatenressource generiert, die Laufzeitdetails verfolgt.

Container-Images verfügen über einen zusätzlichen Workflow, der während der Verteilung ausgeführt wird.

Recherchieren von Details zu Laufzeit-Instance-Fehlern für Ihren Workflow

Um einen Laufzeitfehler für Ihren Workflow zu beheben, können Sie die - GetWorkflowExecution und ListWorkflowStepExecutions-API-Aktionen mit Ihrem aufrufen`workflow execution ID`.

Überprüfen von Workflow-Laufzeitprotokollen

- Amazon CloudWatch -Protokolle

  Image Builder veröffentlicht detaillierte Workflow-Ausführungsprotokolle in der folgenden Image-Builder- CloudWatch Protokollgruppe und im folgenden Stream:

  LogGroup:

  ```
  /aws/imagebuilder/ImageName
  ```

  LogStream (x.x.x/x):

  ```
  ImageVersion/ImageBuildVersion
  ```

  Mit - CloudWatch Protokollen können Sie Protokolldaten mit Filtermustern durchsuchen. Weitere Informationen finden Sie unter Suchen von Protokolldaten mithilfe von Filtermustern im Amazon-CloudWatch Logs-Benutzerhandbuch.

- AWS CloudTrail

  Alle Build-Aktivitäten werden auch angemeldet CloudTrail , wenn sie in Ihrem Konto aktiviert sind. Sie können CloudTrail Ereignisse nach der Quelle filtern`imagebuilder.amazonaws.com`. Alternativ können Sie nach der Amazon EC2-Instance-ID suchen, die im Ausführungsprotokoll zurückgegeben wird, um weitere Details zur Pipeline-Ausführung anzuzeigen.

- Amazon Simple Storage Service (S3)

  Wenn Sie in Ihrer Infrastrukturkonfiguration einen S3-Bucket-Namen und ein Schlüsselpräfix angegeben haben, folgt der Laufzeitprotokollpfad des Workflow-Schritts diesem Muster:

  ```
  S3://S3BucketName/KeyPrefix/ImageName/ImageVersion/ImageBuildVersion/
  WorkflowExecutionId/StepName
  ```

  Die Protokolle, die Sie an Ihren S3-Bucket senden, zeigen die Schritte und Fehlermeldungen für Aktivitäten auf der EC2-Instance während des Image-Build-Prozesses an. Die Protokolle enthalten Protokollausgaben vom Komponentenmanager, die Definitionen der ausgeführten Komponenten und die detaillierte Ausgabe (in JSON) aller auf der Instance durchgeführten Schritte. Wenn Sie auf ein Problem stoßen, sollten Sie diese Dateien, beginnend mit , überprüfen`application.log`, um die Ursache des Problems auf der Instance zu diagnostizieren.

Standardmäßig fährt Image Builder die Amazon EC2-Build- oder Test-Instance herunter, die ausgeführt wird, wenn die Pipeline fehlschlägt. Sie können die Instance-Einstellungen für die Infrastrukturkonfigurationsressource ändern, die Ihre Pipeline verwendet, um Ihre Build- oder Test-Instance zur Fehlerbehebung beizubehalten.

Um die Instance-Einstellungen in der Konsole zu ändern, müssen Sie das Kontrollkästchen Instance bei Fehler beenden deaktivieren, das sich im Abschnitt Fehlerbehebungseinstellungen Ihrer Infrastrukturkonfigurationsressource befindet.

Sie können die Instance-Einstellungen auch mit dem update-infrastructure-configuration Befehl in der ändernAWS CLI. Setzen Sie den `terminateInstanceOnFailure` Wert `false` in der JSON-Datei, auf die der Befehl mit dem `--cli-input-json` Parameter verweist, auf . Details hierzu finden Sie unter Aktualisieren einer Infrastrukturkonfiguration.

Fehlerbehebungsszenarien

In diesem Abschnitt werden die folgenden detaillierten Problembehandlungsszenarien aufgeführt:

- Zugriff verweigert – Statuscode 403
- Build-Timeout bei der Überprüfung der Verfügbarkeit des Systems Manager Agent auf der Build-Instance
- Die sekundäre Windows-Festplatte ist beim Start offline
- Build schlägt mit CIS-gehärtetem Basis-Image fehl
- AssertInventoryCollection schlägt fehl (Systems Manager Automation)

Um die Details eines Szenarios anzuzeigen, wählen Sie den Titel des Szenarios aus, um es zu erweitern. Sie können mehrere Titel gleichzeitig erweitern.

Zugriff verweigert – Statuscode 403

Beschreibung

Der Pipeline-Build schlägt mit dem Statuscode „AccessDenied: Zugriff verweigert: 403“ fehl.

Ursache

Mögliche Gründe hierfür sind:

- Das Instance-Profil verfügt nicht über die erforderlichen Berechtigungen für den Zugriff auf APIs oder Komponentenressourcen.
- Der Instance-Profilrolle fehlen Berechtigungen, die für die Protokollierung in Amazon S3 erforderlich sind. In der Regel tritt dies auf, wenn die Instance-Profilrolle keine PutObject Berechtigungen für Ihre S3-Buckets hat.

Lösung

Abhängig von der Ursache kann dieses Problem wie folgt behoben werden:

- Instance-Profil fehlen -verwaltete Richtlinien – Fügen Sie die fehlenden Richtlinien zu Ihrer Instance-Profilrolle hinzu. Führen Sie dann die Pipeline erneut aus.
- Instance-Profil fehlen Schreibberechtigungen für S3-Bucket – Fügen Sie Ihrer Instance-Profilrolle eine Richtlinie hinzu, die PutObject Berechtigungen zum Schreiben in Ihren S3-Bucket gewährt. Führen Sie dann die Pipeline erneut aus.

Build-Timeout bei der Überprüfung der Verfügbarkeit des Systems Manager Agent auf der Build-Instance

Beschreibung

Der Pipeline-Build schlägt mit „Status = „TimedOut“ und „Fehlermeldung = „Schritt ist abgelaufen, während der Schritt die Verfügbarkeit des Systems Manager Agent auf der/den Ziel-Instance(s) überprüft“.

Ursache

Mögliche Gründe hierfür sind:

- Die Instance, die gestartet wurde, um die Build-Operationen und Komponenten auszuführen, konnte nicht auf den Systems Manager-Endpunkt zugreifen.
- Das Instance-Profil verfügt nicht über die erforderlichen Berechtigungen.

Lösung

Abhängig von der möglichen Ursache kann dieses Problem wie folgt behoben werden:

- Zugriffsproblem, privates Subnetz – Wenn Sie in einem privaten Subnetz erstellen, stellen Sie sicher, dass Sie PrivateLink Endpunkte für Systems Manager, Image Builder und, wenn Sie Protokollierung wünschen, Amazon S3/ eingerichtet habenCloudWatch. Weitere Informationen zum Einrichten von PrivateLink Endpunkten finden Sie unter Konzepte für VPC-Endpunkte (AWS PrivateLink).
- Fehlende Berechtigungen – Fügen Sie Ihrer serviceverknüpften IAM-Rolle für Image Builder die folgenden verwalteten Richtlinien hinzu:
  - EC2InstanceProfileForImageBuilder
  - EC2InstanceProfileForImageBuilderECRContainerBuilds
  - AmazonSSMManagedInstanceCore

  Weitere Informationen zur serviceverknüpften Rolle von Image Builder finden Sie unter Verwenden von serviceverknüpften Rollen für EC2 Image Builder.

Die sekundäre Windows-Festplatte ist beim Start offline

Beschreibung

Wenn der Instance-Typ, der zum Erstellen eines Windows-AMI von Image Builder verwendet wird, nicht mit dem Instance-Typ übereinstimmt, der zum Starten vom AMI verwendet wird, kann ein Problem auftreten, bei dem Nicht-Root-Volumes beim Start offline sind. Dies geschieht hauptsächlich, wenn die Build-Instance eine neuere Architektur als die Start-Instance verwendet.

Das folgende Beispiel zeigt, was passiert, wenn ein Image-Builder-AMI auf einem EC2-Nitro-Instance-Typ erstellt und auf einer EC2-Xen-Instance gestartet wird:

Build-Instance-Typ : m5.large (Nitro)

Start-Instance-Typ : t2.medium (Xen)

```
PS C:\Users\Administrator> get-disk
Number  Friendly Name  Serial Number         Health Status  Operational Status  Total
 Size  Partition Style
------  -------------  -------------         -------------  ------------------
 ----------  ---------------
0       AWS PVDISK     vol0abc12d34e567f8a9  Healthy        Online                  30
 GB  MBR
1       AWS PVDISK     vol1bcd23e45f678a9b0  Healthy        Offline                  8
 GB  MBR
```

Ursache

Aufgrund der Windows-Standardeinstellungen werden neu erkannte Datenträger nicht automatisch online geschaltet und formatiert. Wenn der Instance-Typ auf EC2 geändert wird, behandelt Windows dies als neue Datenträger, die erkannt werden. Dies liegt an der zugrunde liegenden Treiberänderung.

Lösung

Wir empfehlen Ihnen, beim Erstellen Ihres Windows-AMI dasselbe System von Instance-Typen zu verwenden, von dem aus Sie starten möchten. Schließen Sie keine Instance-Typen ein, die auf verschiedenen Systemen in Ihrer Infrastrukturkonfiguration basieren. Wenn einer der von Ihnen angegebenen Instance-Typen das Nitro-System verwendet, sollten sie alle das Nitro-System verwenden.

Weitere Informationen zu Instances, die auf dem Nitro-System basieren, finden Sie unter Instances, die auf dem Nitro-System basieren im Amazon EC2-Benutzerhandbuch für Windows-Instances.

Build schlägt mit CIS-gehärtetem Basis-Image fehl

Beschreibung

Sie verwenden ein CIS-gehärtetes Basis-Image und der Build schlägt fehl.

Ursache

Wenn das `/tmp` Verzeichnis als klassifiziert ist`noexec`, kann dies dazu führen, dass Image Builder fehlschlägt.

Lösung

Wählen Sie im `workingDirectory` Feld des Image-Rezepts einen anderen Speicherort für Ihr Arbeitsverzeichnis aus. Weitere Informationen finden Sie in der Beschreibung des ImageRecipe Datentyps.

AssertInventoryCollection schlägt fehl (Systems Manager Automation)

Beschreibung

Systems Manager Automation zeigt einen Fehler im `AssertInventoryCollection` Automatisierungsschritt an.

Ursache

Möglicherweise haben Sie oder Ihre Organisation eine Systems Manager State Manager-Zuordnung erstellt, die Bestandsinformationen für EC2-Instances sammelt. Wenn die erweiterte Image-Metadatensammlung für Ihre Image-Builder-Pipeline aktiviert ist (dies ist die Standardeinstellung), versucht Image Builder, eine neue Bestandszuordnung für die Build-Instance zu erstellen. Systems Manager lässt jedoch nicht mehrere Bestandszuordnungen für verwaltete Instances zu und verhindert eine neue Zuordnung, falls bereits eine vorhanden ist. Dies führt dazu, dass der Vorgang fehlschlägt und zu einem fehlgeschlagenen Pipeline-Build führt.

Lösung

Um dieses Problem zu beheben, deaktivieren Sie die erweiterte Erfassung von Image-Metadaten mit einer der folgenden Methoden:

- Aktualisieren Sie Ihre Image-Pipeline in der Konsole, um das Kontrollkästchen Erweiterte Metadatensammlung aktivieren zu deaktivieren. Speichern Sie Ihre Änderungen und führen Sie einen Pipeline-Build aus.

  Weitere Informationen zum Aktualisieren Ihrer AMI-Image-Pipeline mithilfe der EC2 Image Builder-Konsole finden Sie unter Aktualisieren von AMI-Image-Pipelines (Konsole). Weitere Informationen zum Aktualisieren Ihrer Container-Image-Pipeline mithilfe der EC2 Image Builder-Konsole finden Sie unter Aktualisieren einer Container-Image-Pipeline (Konsole).
- Sie können Ihre Image-Pipeline auch mit dem update-image-pipeline Befehl in der aktualisierenAWS CLI. Fügen Sie dazu die `-EnhancedImageMetadataEnabled`Eigenschaft in Ihre JSON-Datei ein und setzen Sie auf `false`. Das folgende Beispiel zeigt die Eigenschaft , die auf festgelegt ist`false`.

```
{
    "name": "MyWindows2019Pipeline",
    "description": "Builds Windows 2019 Images",
    "enhancedImageMetadataEnabled": false,
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-infrastructure-configuration",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-distribution-configuration",
```

```
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 60
    },
    "schedule": {
        "scheduleExpression": "cron(0 0 * * SUN *)",
        "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "ENABLED"
}
```

Um dies bei neuen Pipelines zu verhindern, deaktivieren Sie das Kontrollkästchen Erweiterte Metadatensammlung aktivieren, wenn Sie eine neue Pipeline mit der EC2 Image Builder-Konsole erstellen, oder setzen Sie den Wert der `EnhancedImageMetadataEnabled` Eigenschaft in Ihrer JSON-Datei auf , `false` wenn Sie Ihre Pipeline mit der erstellenAWS CLI.

Dokumentverlauf für das EC2 Image Builder-Benutzerhandbuch

In der folgenden Tabelle werden wichtige Änderungen an der Dokumentation nach Datum beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- API-Version: 2023-12-12

Änderung	Beschreibung	Datum
Feature-Version: Image-Workflow-Verwaltung	Mit Image-Workflows haben Sie mehr Flexibilität, Transparenz und Kontrolle über den Prozess der Image-Erstellung. Sie können die Erstellungs- und Testschritte für Ihre Workflows anpassen oder den Image-Builder-Standardworkflow verwenden.	12. Dezember 2023
Feature-Version: Image-Lebenszyklusmanagement	Mit Richtlinien und Regeln zur Verwaltung des Image-Lebenszyklus können Sie Ihre Ressourcenverwaltungsstrategie definieren, um sicherzustellen, dass veraltete Images und die zugehörigen Ressourcen markiert und entfernt werden.	17. November 2023
STIG-Q3-Updates	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des dritten Quartals 2023. Das Messaging wurde aktualisi	05. Oktober 2023

	ert, um zu verdeutlichen, dass Pakete von Drittanbietern nicht automatisch installiert werden, mit sehr wenigen Ausnahmen . Alle übersprungenen STIGs werden protokolliert.	
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des zweiten Quartals 2023.	3. Mai 2023
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des ersten Quartals 2023. Unterstützung für AL2023 hinzugefügt.	14. April 2023
Aktualisieren unterstützter Regionen für AWSTOE	AWSTOE Unterstützung für die folgenden hinzugefü gtAWS-Regionen: Asien-Paz ifik (Hyderabad), Asien-Paz ifik (Jakarta), Europa (Zürich), Europa (Spanien) und Naher Osten (VAE).	13. April 2023
AWSTOE Anwendungs-Download-Updates	Die Signatur für den AWSTOE Installations-Download unter Windows wurde aktualisiert. Beachten Sie auch, dass Anwendungs-Downloads von S3-Buckets jetzt TLS Version 1.2 oder höher erfordern.	31. März 2023

Feature-Version: Erweiterte Build-Workflows	Laufzeitdetails für Image-Builds wurden auf der neuen Workflow-Registerkarte in den Details der Image-Build-Version hinzugefügt. Verbesserte Informationen zur Fehlerbehebung bei Builds.	30. März 2023
Feature-Version: CVE-Erkennung und -Berichterstellung	Für Konten, die Amazon Inspector-Scans aktiviert haben, kann Image Builder die Ergebnisse der allgemeinen Schwachstellen und Risiken (CVE) von Amazon Inspector während der Testphase des Erstellungsprozesses für neue Images erfassen, einschließlich Container-Images, die in Amazon ECR gespeichert sind. Image Builder erstellt einen Snapshot der Ergebnisse, um die Detailanalyse zu unterstützen. Image Builder berichtet auch über die Anzahl der Erkenntnisse, die nach Konto, Pipeline oder Image gefiltert werden können, mit der Möglichkeit, Details detailliert anzuzeigen.	30. März 2023
Versionsverlauf hinzugefügt	Versionsverlauf zu den Windows- und Linux-Abschnitten hinzugefügt.	17. Februar 2023

Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des vierten Quartals 2022.	1. Februar 2023
Feature-Version: AWS Marketplace Integration und CIS-Hardening	AWS Marketplace Integration hinzugefügt, um ein abonniertes Image einfach als Grundlage für ein neues benutzerdefiniertes Image zu finden und zu verwenden, einschließlich CIS-gehärteter Images und einer neuen CIS-Hardening-Komponente aus dem Center for Internet Security.	13. Januar 2023
CIS-Hardening-Komponenten	CIS-Hardening-Komponenten hinzugefügt, die CIS gehören und von CIS verwaltet werden.	13. Januar 2023
Neue STIG-Versionen	Einführung der Ubuntu-Unterstützung, Aktualisierung der STIG-Versionen und Anwendung von STIGS für das zweite Quartal 2022.	20. Juli 2022
Dokumentaktualisierung: Seite „Navigation für YAML-Komponentendokument erstellen“	Der Inhalt des YAML-Komponentendokuments wurde auf eine eigene Seite verschoben und andere Seiten wurden aktualisiert, um darauf zu verweisen.	7. Juni 2022

Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des ersten Quartals 2022.	25. April 2022
ExecuteDocument Aktionsmodul hinzugefügt	Dokumentation für das `ExecuteDocument` Aktionsmodul unter hinzugefügt`General execution` .	28. März 2022
Feature-Version: Unterstützung für schnelleres Starten von Windows AMI	Es wurden Verteilungskonfigurationseinstellungen hinzugefügt, um schnelleres Starten für Windows-AMIs zu unterstützen.	21. Februar 2022
Wartungsversion: Aktualisieren des AWSTOE binären Thumbprints	Der binäre Thumbprint für das AWSTOE Unterzeichnerzertifikat wurde aktualisiert.	18. Februar 2022
Feature-Version: Konfigurieren der Eingabe für AWSTOE	Unterstützung für die Verwendung einer JSON-Konfigurationsdatei als Eingabe für den AWSTOE run Befehl hinzugefügt.	3. Februar 2022
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des vierten Quartals 2021. Außerdem wurde ein Abschnitt für neue SCAP Compliance Checker (SCC)-Komponenten hinzugefügt.	22. Dezember 2021

Feature-Version: VM Import/Export (VMIE)-Integration	Unterstützung für den VM-Import über alle Kanäle (Konsole, API/CLI usw.) und für den VM-Export über API/CLI hinzugefügt. Der VM-Export ist derzeit nicht über die Image-Builder-Konsole verfügbar.	20. Dezember 2021
Feature-Version: AMI-Freigabe für AWS Organizations und OUs	Die Verteilungskonfiguration wurde aktualisiert, um Unterstützung für die Freigabe von Ausgabe-AMIs mit AWS Organizations und OUs hinzuzufügen.	24. November 2021
Dokumentaktualisierung: Aktualisieren von Komponentenphasen und -phasen	Erweiterter Inhalt für Komponentenphasen in Image Builder und wie diese mit AWSTOE Komponentenphasen interagieren.	22. September 2021
Dokumentaktualisierung: Hinzufügen von CloudTrail Integrationsinhalten	Überwachungsübersicht und CloudTrail Integrationsinhalte wurden hinzugefügt.	17. September 2021
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des dritten Quartals 2021.	10 September 2021

Feature-Version: Amazon-EventBridge Integration	Es wurde EventBridge Unterstützung hinzugefügt, mit der Sie Image Builder mit Ereignissen aus verwandte n verbinden und Ereignisse basierend auf Regeln initiiere n könnenAWS-Services, die in definiert sind EventBridge.	18. August 2021
Dokumentaktualisierung: AWSTOE Seiten neu anordnen	Aus Gründen der Übersicht lichkeit wurden die AWSTOE Seiten neu angeordnet.	11. August 2021
Feature-Version: Parametri sierte Komponenten und zusätzliche Instance-Konfigura tion	Unterstützung für die Angabe von Parametern zum Anpassen von Komponent en für Rezepte hinzugefü gt. Erweiterte Konfiguration der EC2-Instances, die zum Erstellen und Testen von Images verwendet werden, einschließlich der Möglichke it, Befehle anzugeben, die beim Start ausgeführt werden sollen, und mehr Kontrolle über die Installation und Entfernung des Systems Manager-Agenten.	7. Juli 2021
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS für die Veröffentlichung des zweiten Quartals 2021.	30. Juni 2021
Verbesserung: Markieren von Verbesserungen	Verbessertes Messaging rund um das Markieren von Ressourcen.	25. Juni 2021

Feature-Version: Integration von Startvorlagen	Unterstützung für die Verwendung von Amazon EC2-Startvorlagen für die AMI-Verteilung in den Verteilungseinstellungen hinzugefügt.	7. April 2021
Feature-Version: Verbesserungen beim Container-Build	Unterstützung für die Konfiguration von Blockgerät-Zuweisungen und die Angabe von AMIs, die als Basis-Image für Container-Builds verwendet werden sollen, wurde hinzugefügt.	7. April 2021
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewendete STIGS.	5. März 2021
Cron-Ausdrücke aktualisieren	Die Cron-Verarbeitung von Image Builder wurde aktualisiert, um die Granularität von Cron-Ausdrücken auf die Minute zu erhöhen und eine standardmäßige Cron-Planungs-Engine zu verwenden. Beispiele werden mit dem neuen Format aktualisiert.	8. Februar 2021

Feature-Version: Container-Unterstützung	Unterstützung für die Erstellung von Docker-Container-Images mit Image Builder hinzugefügt, mit Registrierung und Speicherung der resultierenden Images in Amazon Elastic Container Registry (Amazon ECR). Die Inhalte wurden neu angeordnet, um neue Funktionen widerzuspiegeln und dem zukünftigen Wachstum gerecht zu werden.	17. Dezember 2020
Neustrukturierte Cron-Dokumentation	Auf dieser Seite finden Sie jetzt weitere Informationen darüber, wie Cron mit Image-Builder-Pipeline-Builds funktioniert, und Details zur UTC-Zeit. Platzhalter, die für bestimmte Felder nicht zulässig sind, wurden entfernt. Beispiele sind jetzt Ausdrucksbeispiele sowohl für die Konsole als auch für die CLI.	13. November 2020
Konsolenversion 2.0: Pipeline-Bearbeitung aktualisiert	Inhaltsänderungen bei den ersten Schritten und der Erstellung von Pipeline-Tutorials sowie die Seite Image-Pipelines verwalten, um neue Konsolenfunktionen und neuen Flow zu integrieren.	13. November 2020

Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewendete STIGS. Hinweis – Listenformat wurde geändert, um STIGs anzuzeigen, die standardmäßig angewendet werden.	15. Oktober 2020
Unterstützung für Looping-Konstrukte in AWSTOE	Erstellen Sie Schleifenkonstrukte, um eine wiederholte Abfolge von Anweisungen in der AWSTOE Anwendung zu definieren.	29. Juli 2020
Unterstützung für die lokale Entwicklung von AWSTOE Komponenten	Entwickeln und testen Sie Image-Komponenten lokal mit der AWSTOE Anwendung.	28. Juli 2020
Verschlüsselte AMIs	EC2 Image Builder bietet Unterstützung für verschlüsselte AMI-Verteilungen.	1. Juli 2020
AutoScaling Veraltung	Veraltung der Verwendung von AutoScaling zum Starten von Instances.	15. Juni 2020

Unterstützung für Konnektivität über AWS PrivateLink	Sie können eine private Verbindung zwischen Ihrer VPC und EC2 Image Builder herstellen, indem Sie einen Schnittstellen-VPC-Endpunkt erstellen. Schnittstellenendpunkte werden von unterstützt AWS PrivateLink, einer Technologie, mit der Sie ohne Internet-Gateway, NAT-Gerät, VPN-Verbindung oder AWS Direct-Connect-Verbindung privat auf Image-Builder-APIs zugreifen können. Instances in Ihrer VPC benötigen für die Kommunikation mit Image-Builder-APIs keine öffentlichen IP-Adressen. Der Datenverkehr zwischen Ihrer VPC und Image Builder verlässt das Amazon-Netzwerk nicht.	10. Juni 2020
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewendete STIGS.	23. Januar 2020
Fehlersuche	Allgemeine Problembehandlungsszenarien wurden hinzugefügt.	22. Januar 2020
STIG-Komponenten	Sie können STIG-konforme Images mit AWSTOE STIG-Komponenten erstellen.	22. Januar 2020

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.