AWS KMS Kryptografische Details

AWS Key Management Service

AWS Key Management Service: AWS KMS Kryptografische Details

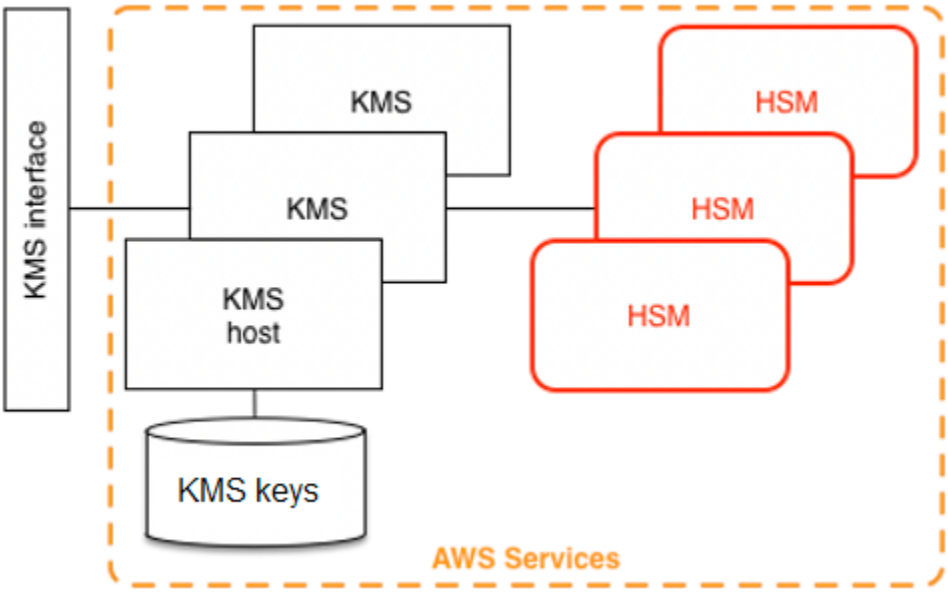
Table of Contents

Einführung in die kryptografischen Details von AWS KMS

AWS Key Management Service (AWS KMS) bietet eine Webschnittstelle zum Generieren und Verwalten von kryptografischen Schlüsseln und fungiert als kryptografischer Service-Anbieter zum Schutz von Daten. AWS KMS bietet traditionelle Schlüsselverwaltungsservices, die mit AWS-Services integriert sind, um eine konsistente Ansicht der Schlüssel der Kunden über AWS, mit zentralisierter Verwaltung und Prüfung zu bieten. Dieses Whitepaper enthält eine detaillierte Beschreibung zu den kryptografischen Operationen von AWS KMS, um Sie bei der Bewertung der vom Service angebotenen Funktionen zu unterstützen.

AWS KMS enthält eine Weboberfläche über die AWS Management Console, eine Befehlszeilenschnittstelle und RESTful-API-Operationen, um kryptografische Operationen einer verteilten Flotte von FIPS 140-2 validierten Hardware-Sicherheitsmodulen (HSMs) anzufordern [1]. Die AWS KMS-HSM ist eine eigenständige Multichip-Hardware-Kryptografie-Appliance, die speziell für die Bereitstellung dedizierter kryptografischer Funktionen zur Erfüllung der Sicherheits- und Skalierbarkeitsanforderungen von AWS KMS erstellt wurden. Sie können Ihre eigene HSM-basierte kryptografische Hierarchie unter Schlüsseln einrichten, die Sie als AWS KMS keys verwaltet werden. Diese Schlüssel werden nur auf den HSMs und nur für die erforderliche Zeit zur Bearbeitung Ihrer kryptografischen Anfrage im Speicher zur Verfügung gestellt. Sie können mehrere KMS-Schlüssel erstellen, die jeweils durch seine Schlüssel-ID dargestellt werden. Nur unter AWS-IAM-Rollen und -Konten, die von jedem Kunden verwaltet werden, können Kunden-KMS-Schlüssel erstellt, gelöscht oder zum Verschlüsseln, Entschlüsseln, Signieren oder Verifizieren von Daten verwendet werden. Sie können Zugriffskontrollen dafür definieren, wer KMS-Schlüssel verwalten und/oder verwenden kann, indem Sie eine Richtlinie erstellen, die an den Schlüssel angehängt ist. Mithilfe dieser Richtlinien können Sie anwendungsspezifische Verwendungen für Ihre Schlüssel für jeden API-Vorgang definieren.

Darüber hinaus unterstützen die meisten AWS-Services die Verschlüsselung von Data-at-Rest mit KMS-Schlüsseln. Mit dieser Funktion können Kunden steuern, wie und wann AWS-Services auf verschlüsselte Daten zugreifen können, indem sie steuern, wie und wann auf KMS-Schlüssel zugegriffen werden kann.



AWS KMS ist ein mehrstufiger Service, der aus AWS KMS-Hosts und einer Reihe von HSMs besteht. Die Gruppierung dieser gestuften Hosts bildet die AWS KMS-Stacks. Alle Anforderungen an AWS KMS müssen über das Transport Layer Security Protocol (TLS) erfolgen und auf einem AWS KMS-Host enden. AWS KMS-Hosts lassen TLS nur mit einer Verschlüsselungssuite zu, die eine perfekte Vorwärtsgeheimhaltung bietet. AWS KMS authentifiziert und autorisiert Ihre Anfragen mit den gleichen Anmelde- und Richtlinienmechanismen von AWS Identity and Access Management (IAM), die für alle anderen AWS-API-Operationen verfügbar sind.

Grundkonzepte

Das Erlernen einiger grundlegender Begriffe und Konzepte wird Ihnen helfen, das Beste aus AWS Key Management Service herauszuholen.

AWS KMS key

> **Note**
>
> AWS KMS ersetzt den Begriff Kundenhauptschlüssel (CMK) durch AWS KMS key und KMS-Schlüssel. Das Konzept hat sich nicht geändert. Um zu verhindern, dass große Veränderungen gemacht werden, behält AWS KMS einige Variationen dieses Begriffs bei.

Ein logischer Schlüssel, der den oberen Teil Ihrer Schlüsselhierarchie darstellt. Einem KMS-Schlüssel wird ein Amazon-Ressourcenname (ARN) zugewiesen, der eine eindeutige Schlüsselkennung oder Schlüssel-ID enthält. AWS KMS keys haben drei Typen:

- Vom Kunden verwalteter Schlüssel – Kunden erstellen und steuern den Lebenszyklus und die wichtigsten Richtlinien von vom Kunden verwalteten Schlüsseln. Alle Anforderungen an diese Schlüssel werden als CloudTrail Ereignisse protokolliert.
- Von AWS verwaltete Schlüssel – AWS erstellt und kontrolliert den Lebenszyklus und die Schlüsselrichtlinien von Von AWS verwaltete Schlüssel, die Ressourcen im AWS-Konto eines Kunden sind. Kunden können Zugriffsrichtlinien und CloudTrail Ereignisse für anzeigenVon AWS verwaltete Schlüssel, jedoch keine Aspekte dieser Schlüssel verwalten. Alle Anforderungen an diese Schlüssel werden als CloudTrail Ereignisse protokolliert.
- AWS-eigene Schlüssel – Diese Schlüssel werden von AWS erstellt und ausschließlich für interne Verschlüsselungsvorgänge über verschiedene AWS-Services hinweg verwendet. Kunden haben keinen Einblick in wichtige Richtlinien oder die AWS-eigener Schlüssel Nutzung in CloudTrail.

Alias

Ein benutzerfreundlicher Name, der einem KMS-Schlüssel zugeordnet ist. Der Alias kann in vielen der AWS KMS-API-Operationen austauschbar mit der Schlüssel-ID verwendet werden.

Berechtigungen

Eine Richtlinie, die einem KMS-Schlüssel zugeordnet ist, die Berechtigungen für den Schlüssel definiert. Die Standardrichtlinie lässt alle von Ihnen definierten Prinzipale zu und erlaubt dem AWS-Konto, IAM-Richtlinien hinzuzufügen, die auf den Schlüssel verweisen.

Gewährungen

Die delegierte Berechtigung zur Verwendung eines KMS-Schlüssels, wenn die beabsichtigten IAM-Prinzipale oder die Nutzungsdauer zu Beginn nicht bekannt sind und daher nicht zu einem Schlüssel oder einer IAM-Richtlinie hinzugefügt werden können. Eine Verwendung von Grants besteht darin, eingeschränkte Berechtigungen dafür zu definieren, wie ein AWS-Service einen KMS-Schlüssel verwenden kann. Der Service muss möglicherweise Ihren Schlüssel verwenden, um in Ihrem Namen asynchrone Arbeiten an verschlüsselten Daten durchzuführen, wenn kein direkt signierter API-Aufruf von Ihnen vorliegt.

Datenschlüssel

Auf HSMs generierte kryptografische Schlüssel, die durch einen KMS-Schlüssel geschützt sind. AWS KMS ermöglicht es autorisierten Entitäten, Datenschlüssel abzurufen, die durch einen

KMS-Schlüssel geschützt sind. Sie können sowohl als Klartext-Datenschlüssel (unverschlüsselt) als auch als verschlüsselte Datenschlüssel zurückgegeben werden. Datenschlüssel können symmetrisch oder asymmetrisch sein (wobei sowohl der öffentliche als auch der private Teil zurückgegeben wird).

Verschlüsselungstexte

Die verschlüsselte Ausgabe von AWS KMS, manchmal auch als Kundenverschlüsselungstext bezeichnet, um Verwirrung zu vermeiden. Verschlüsselungstext enthält verschlüsselte Daten mit zusätzlichen Informationen, die den KMS-Schlüssel identifizieren, der im Entschlüsselungsprozess verwendet werden soll. Verschlüsselte Datenschlüssel sind ein gängiges Beispiel für Verschlüsselungstext, der bei Verwendung eines KMS-Schlüssels erzeugt wird. Alle Daten mit einer Größe von 4 KB können jedoch unter einem KMS-Schlüssel verschlüsselt werden, um einen Verschlüsselungstext zu erzeugen.

Verschlüsselungskontext

Eine Schlüssel-Wert-Paar-Zuordnung mit zusätzlichen Informationen, die mit AWS KMS-geschützten Informationen verknüpft sind. AWS KMS verwendet eine authentifizierte Verschlüsselung, um Datenschlüssel zu schützen. Der Verschlüsselungskontext wird in AWS KMS-verschlüsselten Verschlüsselungstexten in den AAD der authentifizierten Verschlüsselung eingebunden. Diese Kontextinformation ist optional und wird nicht zurückgegeben, wenn Sie einen Schlüssel (oder einen Verschlüsselungsvorgang) anfordern. Dieser Kontextwert ist jedoch erforderlich, um einen Entschlüsselungsvorgang erfolgreich abzuschließen. Eine beabsichtigte Verwendung des Verschlüsselungskontexts besteht darin, zusätzliche authentifizierte Informationen bereitzustellen. Diese Informationen können Ihnen helfen, Richtlinien durchzusetzen und in die AWS CloudTrail Protokolle aufgenommen zu werden. Sie können beispielsweise ein Schlüssel-Wert-Paar von {"key name":"satellite uplink key"} verwenden, um den Datenschlüssel zu benennen. Bei der anschließenden Verwendung des Schlüssels wird ein AWS CloudTrail-Eintrag erstellt, der „Schlüsselname“ enthält: „Satelliten-Uplink-Schlüssel“. Diese zusätzlichen Informationen können einen nützlichen Kontext liefern, um zu verstehen, warum ein bestimmter KMS-Schlüssel verwendet wurde.

Öffentlicher Schlüssel

Wenn asymmetrische Verschlüsselungen (RSA oder elliptische Kurve) verwendet werden, ist der öffentliche Schlüssel die „öffentliche Komponente“ eines öffentlich-privaten Schlüsselpaars. Der öffentliche Schlüssel kann freigegeben und an Entitäten verteilt werden, die Daten für den Besitzer des öffentlich-privaten Schlüsselpaars verschlüsseln müssen. Bei digitalen Signaturvorgängen wird der öffentliche Schlüssel verwendet, um die Signatur zu überprüfen.

Privater Schlüssel

Bei der Verwendung asymmetrischer Verschlüsselungen (RSA oder elliptische Kurve) ist der private Schlüssel die „private Komponente“ eines öffentlich-privaten Schlüsselpaares. Mit dem privaten Schlüssel werden dann Daten entschlüsselt oder digitale Signaturen erstellt. Ähnlich wie symmetrische KMS-Schlüssel werden private Schlüssel in HSMs verschlüsselt. Sie werden nur in das Kurzzeitgedächtnis des HSM und nur für die Zeit entschlüsselt, die für die Bearbeitung Ihrer kryptografischen Anfrage benötigt wird.

AWS KMS Designziele

AWS KMS wurde entwickelt, um die folgenden Anforderungen zu erfüllen.

Haltbarkeit

Die Haltbarkeit von kryptografischen Schlüsseln ist so ausgelegt, dass sie der der Services mit der höchsten Haltbarkeit in AWS entspricht. Ein einzelner kryptografischer Schlüssel kann große Datenmengen verschlüsseln, die sich über eine lange Zeit angesammelt haben.

Vertrauenswürdig

Die Verwendung von Schlüsseln ist durch Zugriffssteuerungsrichtlinien geschützt, die Sie definieren und verwalten. Es gibt keinen Mechanismus, um Klartext-KMS-Schlüssel zu exportieren. Die Vertraulichkeit Ihrer kryptografischen Schlüssel ist von entscheidender Bedeutung. Mehrere Amazon-Mitarbeiter mit rollenspezifischem Zugriff auf quorumbasierte Zugriffskontrollen sind erforderlich, um Verwaltungsaktionen an den HSMs durchzuführen.

Niedrige Latenz und hoher Durchsatz

AWS KMS bietet kryptografische Operationen mit Latenz- und Durchsatzstufen, die für die Verwendung durch andere Services in AWS geeignet sind.

Unabhängige Regionen

AWS bietet unabhängige Regionen für Kunden, die den Datenzugriff in verschiedenen Regionen einschränken müssen. Die Schlüsselverwendung kann innerhalb eines AWS-Region isoliert werden.

Sichere Quelle von Zufallszahlen

Da starke Kryptographie von einer wirklich unvorhersehbaren Zufallszahlengenerierung abhängt, bietet AWS KMS eine qualitativ hochwertige und validierte Quelle für Zufallszahlen.

Audit

AWS KMS zeichnet die Verwendung und Verwaltung kryptografischer Schlüssel in -AWS CloudTrail Protokollen auf. Sie können AWS CloudTrail-Protokolle verwenden, um die Verwendung Ihrer kryptografischen Schlüssel zu überprüfen, einschließlich der Verwendung von Schlüsseln durch AWS-Services in Ihrem Namen.

Um diese Ziele zu erreichen, umfasst das AWS KMS-System eine Reihe von AWS KMS-Operatoren und Service-Host-Operatoren (zusammen „Operatorenr“), die „Domänen“ verwalten. Eine Domäne ist ein regional definierter Satz von AWS KMS-Servern, HSMs und Operatoren. Jeder AWS KMS-Operator verfügt über ein Hardware-Token, das ein privates und öffentliches Schlüsselpaar enthält, das zur Authentifizierung seiner Aktionen verwendet wird. Die HSMs verfügen über ein zusätzliches privates und öffentliches Schlüsselpaar zum Einrichten von Verschlüsselungsschlüsseln, die die HSM-Statussynchronisierung schützen.

Dieses Dokument veranschaulicht, wie AWS KMS Ihre Schlüssel und andere Daten schützt, die Sie verschlüsseln möchten. In diesem Dokument werden Verschlüsselungsschlüssel oder Daten, die Sie verschlüsseln möchten, als „Geheimnisse“ oder „Geheimmaterial“ bezeichnet.

AWS Key Management Service Grundlagen

Die Themen in diesem Kapitel beschreiben die kryptographischen Primitive von AWS Key Management Service und wo sie verwendet werden. Sie führen auch die Grundelemente von AWS KMS ein.

Themen

Kryptografische Primitive

AWS KMS verwendet konfigurierbare kryptografische Algorithmen, sodass das System schnell von einem genehmigten Algorithmus oder Modus zu einem anderen migrieren kann. Der anfängliche Standardsatz kryptografischer Algorithmen wurde aufgrund ihrer Sicherheitseigenschaften und Leistung aus den vom Federal-Information-Processing-Standard-(FIPS)-geprüften Algorithmen ausgewählt.

Entropie und Zufallszahlengenerierung

AWS KMS-Schlüsselgenerierung wird auf den AWS KMS-HSMs durchgeführt. Die HSMs implementieren einen hybriden Zufallszahlengenerator, der den NIST SP800-90A Deterministic Random Bit Generator (DRBG) CTR_DRBG unter Verwendung von AES-256 verwendet. Es wird mit einem nicht-deterministischen Zufallsbitgenerator mit 384 Bit Entropie gesät und mit zusätzlicher Entropie aktualisiert, um Prognosewiderstand bei jedem Aufruf von kryptografischem Material zu bieten.

Symmetrische Schlüsseloperationen (nur Verschlüsselung)

Alle in HSMs verwendeten Verschlüsselungsbefehle mit symmetrischen Schlüsseln verwenden die Advanced Encryption Standards (AES) im Galois Counter Mode (GCM) mit 256-Bit-Schlüsseln. Die analogen Aufrufe zum Entschlüsseln verwenden die umgekehrte Funktion.

AES-GCM ist ein authentifiziertes Verschlüsselungsschema. Neben der Verschlüsselung von Klartext zur Erzeugung von Verschlüsselungstext berechnet es ein Authentifizierungs-Tag über den Verschlüsselungstext und alle zusätzlichen Daten, für die eine Authentifizierung erforderlich ist

(zusätzlich authentifizierte Daten oder AAD). Mit dem Authentifizierungs-Tag wird sichergestellt, dass die Daten aus der angeblichen Quelle stammen und dass der Verschlüsselungstext und AAD nicht geändert wurden.

In unseren Beschreibungen lässt AWS häufig die Einbeziehung des AAD aus, insbesondere wenn es um die Verschlüsselung von Datenschlüsseln geht. Durch umgebenden Text wird in diesen Fällen impliziert, dass die zu verschlüsselnde Struktur zwischen dem zu verschlüsselnden Klartext und dem zu schützenden Klartext-AAD aufgeteilt ist.

AWS KMS bietet Ihnen die Möglichkeit, Schlüsselmaterial in ein AWS KMS key zu importieren, anstatt sich darauf zu verlassen, dass AWS KMS den Schlüssel generiert. Dieses importierte Schlüsselmaterial kann mit RSAES-OAEP oder RSAES-PKCS1-v1_5 verschlüsselt werden, um den Schlüssel beim Transport zum AWS KMS-HSM zu schützen. Die RSA-Schlüsselpaare werden auf AWS KMS-HSMs generiert. Das importierte Schlüsselmaterial wird auf einem AWS KMS-HSM entschlüsselt und unter AES-GCM erneut verschlüsselt, bevor es vom Service gespeichert wird.

Asymmetrische Schlüsseloperationen (Verschlüsselung, digitale Signatur und Signaturüberprüfung)

AWS KMS unterstützt die Verwendung asymmetrischer Schlüsseloperationen sowohl für Verschlüsselung als auch für digitale Signaturvorgänge. Asymmetrische Schlüsselvorgänge basieren auf einem mathematisch verwandten Paar aus öffentlichem Schlüssel und privatem Schlüssel, das Sie zum Ver- und Entschlüsseln oder Signieren und Verifizieren von Signaturen verwenden können, aber nicht beides. Der private Schlüssel verlässt AWS KMS niemals unverschlüsselt. Sie können den öffentlichen Schlüssel innerhalb von AWS KMS verwenden, indem Sie die AWS KMS-API-Operationen aufrufen oder den öffentlichen Schlüssel herunterladen und ihn außerhalb von AWS KMS verwenden.

AWS KMS unterstützt zwei Arten von asymmetrischen Verschlüsselungen.

- RSA-OAEP (zur Verschlüsselung) und RSA-PSS und RSA-PKCS-#1-v1_5 (zum Signieren und Verifizieren) – Unterstützt RSA-Schlüssellängen (in Bit): 2048, 3072 und 4096 für unterschiedliche Sicherheitsanforderungen.
- Elliptische Kurve (ECC) – Wird ausschließlich für Signatur und Verifizierung verwendet. Unterstützt ECC-Kurven: NIST P256, P384, P521, SECP 256k1.

Schlüsselableitungsfunktionen

Eine Schlüsselableitungsfunktion wird verwendet, um zusätzliche Schlüssel aus einem anfänglichen Geheimnis oder Schlüssel abzuleiten. AWS KMS verwendet eine Schlüsselableitungsfunktion (Key derivation function, KDF), um Schlüssel pro Anruf für jede Verschlüsselung unter einem AWS KMS key abzuleiten. Alle KDF-Operationen verwenden die KDF im Zählermodus mit HMAC [FIPS197] mit SHA256 [FIPS180]. Der abgeleitete 256-Bit-Schlüssel wird mit AES-GCM verwendet, um Kundendaten und Schlüssel zu verschlüsseln oder zu entschlüsseln.

AWS KMS interne Verwendung digitaler Signaturen

Digitale Signaturen werden auch verwendet, um Befehle und Kommunikationen zwischen AWS KMS Entitäten zu authentifizieren. Alle Dienstentitäten verfügen über ein Schlüsselpaar des elliptischen Kurvendigitalsignaturalgorithmus (ECDSA). Sie führen ECDSA gemäß der Definition in: Verwendung von Algorithmen der Elliptic Curve Cryptography (ECC) in der kryptografischen Nachrichtensyntax (CMS) und X9.62-2005: Kryptografie mit öffentlichen Schlüsseln für die Finanzdienstleistungsbranche: Der Elliptic Curve Digital Signature Algorithm (ECDSA) durch. Die Entitäten verwenden den sicheren Hash-Algorithmus, der in Publikationen der Bundesinformationsverarbeitungs standards, FIPS PUB 180-4, bekannt als SHA384, definiert ist. Die Schlüssel werden auf der Kurve secp384r1 (NIST-P384) generiert.

Envelope-Verschlüsselung

Eine grundlegende Konstruktion, die in vielen kryptographischen Systemen verwendet wird, ist die Envelope-Verschlüsselung. Bei der Envelope-Verschlüsselung werden zwei oder mehr kryptografische Schlüssel verwendet, um eine Nachricht zu sichern. Typischerweise wird ein Schlüssel von einem längerfristigen statischen Schlüssel k abgeleitet, und ein anderer Schlüssel ist ein Schlüssel pro Nachricht, msgKey, der erzeugt wird, um die Nachricht zu verschlüsseln. Der Umschlag (Envelope) wird durch Verschlüsseln der Nachricht gebildet: ciphertext = Encrypt(msgKey, message) . Anschließend wird der Nachrichtenschlüssel mit dem statischen Langzeitschlüssel verschlüsselt: encKey = Encrypt(k, msgKey) . Schließlich werden die beiden Werte (encKey, Ciphertext) in eine einzelne Struktur oder eine Envelope-veschlüsselte Nachricht verpackt.

Der Empfänger mit Zugriff auf k kann die Envelope-Nachricht öffnen, indem er zuerst den verschlüsselten Schlüssel entschlüsselt und dann die Nachricht entschlüsselt.

AWS KMS bietet die Möglichkeit, diese längerfristigen statischen Schlüssel zu verwalten und den Prozess der Envelope-Verschlüsselung Ihrer Daten zu automatisieren.

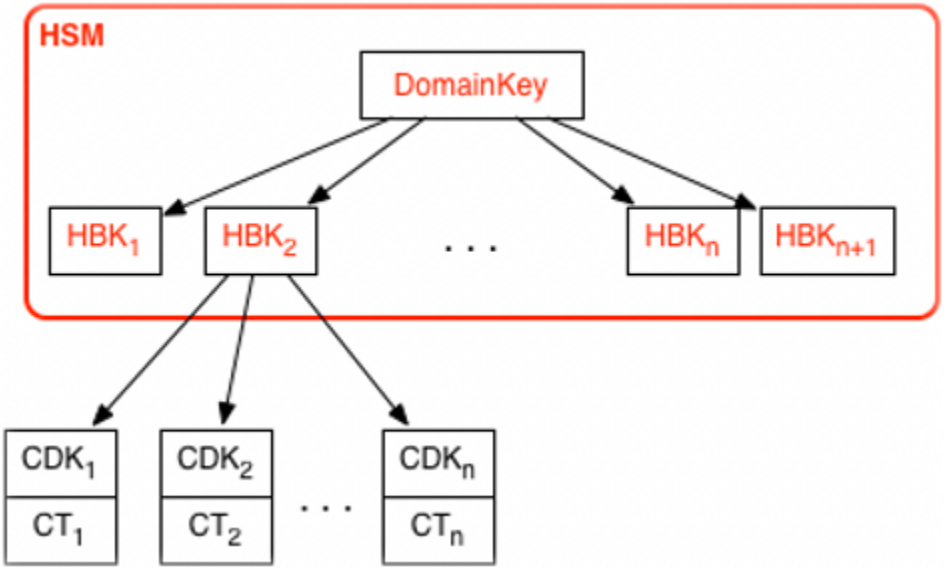
Zusätzlich zu den im AWS KMS-Service bereitgestellten Verschlüsselungsfunktionen bietet das AWS-Verschlüsselungs-SDK clientseitige Envelope-Verschlüsselungsbibliotheken. Sie können diese Bibliotheken verwenden, um Ihre Daten und die Verschlüsselungsschlüssel zu schützen, die zum Verschlüsseln dieser Daten verwendet werden.

AWS KMS key-Hierarchie

Ihre Schlüsselhierarchie beginnt mit einem logischen Schlüssel der obersten Ebene, einem AWS KMS key. Ein KMS-Schlüssel stellt einen Container für Schlüsselmaterial der obersten Ebene dar und ist innerhalb des AWS-Service-Namespace mit einem Amazon-Ressourcennamen (ARN) eindeutig definiert. Der ARN enthält eine eindeutig generierte Schlüsselkennung, eine Schlüssel-ID. Ein KMS-Schlüssel wird basierend auf einer vom Benutzer initiierten Anforderung über AWS KMS erstellt. Beim Empfang fordert AWS KMS die Erstellung eines anfänglichen HSM-Unterstützungsschlüssel (HSM Backing Key, HBK) an, der in den KMS-Schlüsselcontainer platziert wird. Der HBK wird auf einem HSM in der Domäne generiert und ist so konzipiert, dass er niemals im Klartext aus dem HSM exportiert wird. Stattdessen wird der HBK unter HSM-verwalteten Domänenschlüsseln verschlüsselt exportiert. Diese exportierten HBKs werden als exportierte Schlüsseltoken (EKTs) bezeichnet.

Der EKT wird in einen äußerst langlebigen Speicher mit geringer Latenz exportiert. Angenommen, Sie erhalten einen ARN für den logischen KMS-Schlüssel. Dies stellt für Sie die Spitze einer Schlüsselhierarchie oder eines kryptografischen Kontexts dar. Sie können in Ihrem Konto mehrere KMS-Schlüssel erstellen und Richtlinien für Ihre KMS-Schlüssel wie für jede andere AWS benannte Ressource festlegen.

Innerhalb der Hierarchie eines bestimmten KMS-Schlüssels kann man sich den HBK als eine Version des KMS-Schlüssels vorstellen. Wenn Sie den KMS-Schlüssel durch AWS KMS drehen möchten, wird ein neuer HBK erstellt und dem KMS-Schlüssel als aktiver HBK für den KMS-Schlüssel zugeordnet. Die älteren HBKs bleiben erhalten und können verwendet werden, um zuvor geschützte Daten zu entschlüsseln und zu überprüfen. Aber nur der aktive kryptografische Schlüssel kann verwendet werden, um neue Informationen zu schützen.



Sie können über AWS KMS Anfragen stellen, um Ihre KMS-Schlüssel zum direkten Schutz von Informationen zu verwenden, oder zusätzliche HSM-generierte Schlüssel anfordern, die unter Ihrem KMS-Schlüssel geschützt sind. Diese Schlüssel werden als Kundendatenschlüssel oder CDKs bezeichnet. CDKs können verschlüsselt als Verschlüsselungstext (CT), in Klartext oder beides zurückgegeben werden. Alle unter einem KMS-Schlüssel verschlüsselten Objekte (entweder vom Kunden bereitgestellte Daten oder vom HSM generierte Schlüssel) können nur auf einem HSM über einen Aufruf über AWS KMS entschlüsselt werden.

Der zurückgegebene Verschlüsselungstext oder die entschlüsselte Nutzlast wird niemals innerhalb von AWS KMS gespeichert. Die Informationen werden Ihnen über Ihre TLS-Verbindung an AWS KMS zurückgesendet. Dies gilt auch für Anrufe, die von AWS-Services in Ihrem Auftrag getätigt werden.

Die Schlüsselhierarchie und die spezifischen Schlüsseleigenschaften werden in der folgenden Tabelle angezeigt.

Schlüssel	Beschreibung	Lebenszyklus
Domain-Schlüssel	Ein 256-Bit-AES-GCM-Schlüssel nur im Speicher eines HSM, der verwendet wird, um Versionen der KMS-Schlüssel, die HSM-Unter stützungsschlüssel einzuschließen.	Täglich gedreht [1]

Schlüssel	Beschreibung	Lebenszyklus
HSM-Unterstützungsschlüssel	Ein symmetrischer 256-Bit-Schlüssel oder privater RSA- oder elliptischer Kurvenschlüssel, der zum Schutz von Kundendaten und -schlüsseln verwendet und unter Domänensc hlüsseln verschlüsselt gespeichert wird. Ein oder mehrere HSM-Unter stützungsschlüssel umfassen den KMS-Schlüssel, dargestellt durch die keyId.	Jährlich gedreht [2] (optionale Konfiguration)
Abgeleiteter Verschlüsselungssc hlüssel	Ein 256-Bit-AES-GCM-Schlüssel nur im Speicher eines HSM, der zum Verschlüsseln von Kundendaten und Schlüsseln verwendet wird. Abgeleite t von einem HBK für jede Verschlüs selung.	Wird einmal pro Verschlüsselung verwendet und beim Entschlüs seln regeneriert
Kunden-Datenschlüssel	Benutzerdefinierter symmetrischer oder asymmetrischer Schlüssel, der aus HSM in Klartext und Verschlüs selungstext exportiert wird. Unter einem HSM-Unterstützungs schlüssel verschlüsselt und über den TLS-Kanal an autorisierte Benutzer zurückgegeben.	Drehung und Nutzung durch Anwendung gesteuert

[1] AWS KMS kann von Zeit zu Zeit die Domänenschlüsselrotation höchstens auf wöchentlich reduzieren, um Domänenverwaltungs- und Konfigurationsaufgaben zu berücksichtigen.

[2] Standard-Von AWS verwaltete Schlüssel, die von AWS KMS in Ihrem Namen erstellt und verwaltet werden, werden automatisch jährlich rotiert.

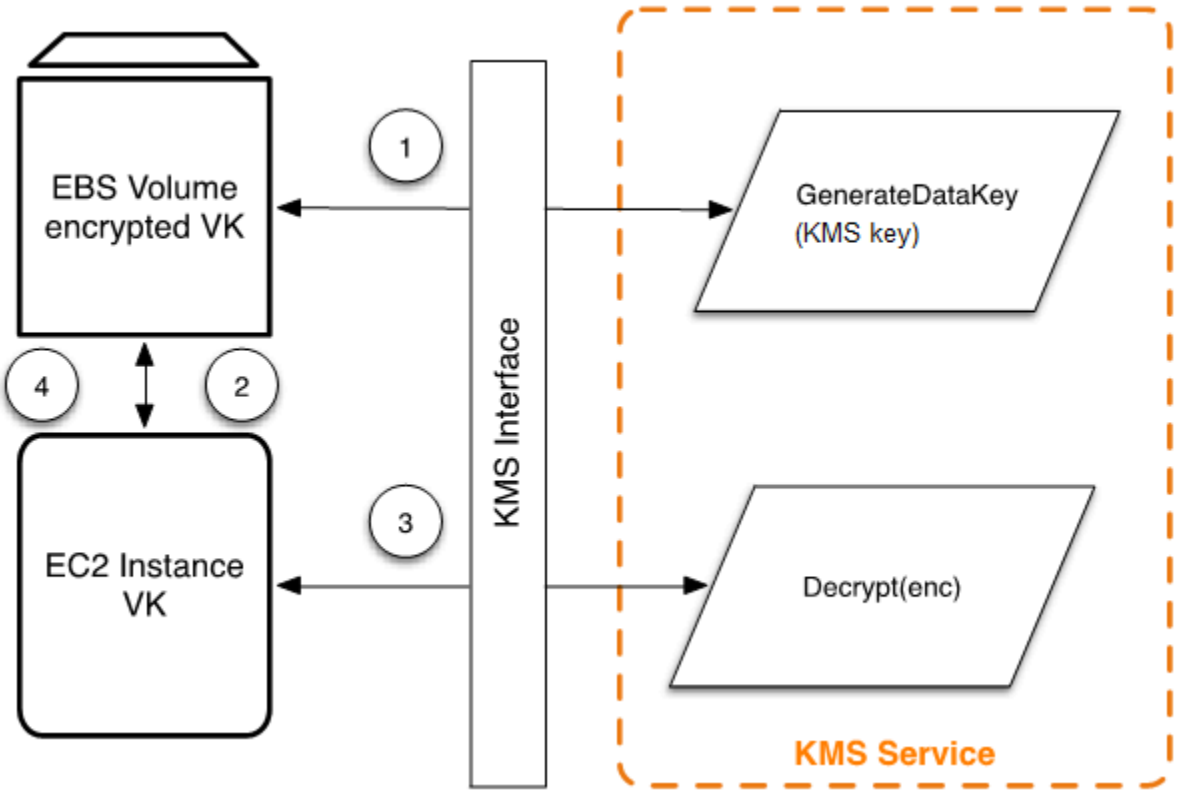
Anwendungsfälle für AWS KMS

Anwendungsfälle können Ihnen dabei helfen, das Beste aus AWS Key Management Service herauszuholen. Die erste zeigt, wie AWS KMS eine serverseitige Verschlüsselung mit AWS KMS keys auf einem Amazon-Elastic-Block-Store-(Amazon-EBS)-Volume durchführt. Die zweite ist eine clientseitige Anwendung, die zeigt, wie Sie die Umschlagverschlüsselung verwenden können, um Inhalte mit AWS KMS zu schützen.

Themen

Amazon-EBS-Volume-Verschlüsselung

Amazon EBS bietet Volume-Verschlüsselungsfunktionen. Jedes Volume wird mithilfe von AES-256-XTS verschlüsselt. Dies erfordert zwei 256-Bit-Volume-Schlüssel, die Sie sich als einen 512-Bit-Volume-Schlüssel vorstellen können. Der Volume-Schlüssel wird unter einem KMS-Schlüssel in Ihrem Konto verschlüsselt. Damit Amazon EBS ein Volume für Sie verschlüsselt, muss es Zugriff haben, um einen Volume-Schlüssel (Volume key, VK) unter einem KMS-Schlüssel im Konto zu generieren. Sie tun dies, indem Sie Amazon EBS für den KMS-Schlüssel gewähren, um Datenschlüssel zu erstellen und diese Volume-Schlüssel zu verschlüsseln und zu entschlüsseln. Jetzt verwendet Amazon EBS AWS KMS mit einem KMS-Schlüssel, um AWS KMS verschlüsselte Volume-Schlüssel zu generieren.



Der folgende Workflow verschlüsselt Daten, die auf ein Amazon-EBS-Volume geschrieben werden:

1. Amazon EBS ruft einen verschlüsselten Volume-Schlüssel unter einem KMS-Schlüssel über AWS KMS über eine TLS-Sitzung ab und speichert den verschlüsselten Schlüssel mit den Volume-Metadaten.
2. Wenn das Amazon-EBS-Volume bereitgestellt wird, wird der verschlüsselte Volume-Schlüssel abgerufen.
3. Ein Aufruf von AWS KMS über TLS erfolgt, um den verschlüsselten Volumenschlüssel zu entschlüsseln. AWS KMS identifiziert den KMS-Schlüssel und stellt eine interne Anfrage an ein HSM in der Flotte, um den verschlüsselten Volume-Schlüssel zu entschlüsseln. AWS KMS gibt den Volume-Schlüssel dann über die TLS-Sitzung an den Amazon-Elastic-Compute-Cloud-(Amazon-EC2)-Host zurück, der Ihre Instance enthält.
4. Der Volume-Schlüssel wird zum Verschlüsseln und Entschlüsseln aller Daten verwendet, die zum und vom angeschlossenen Amazon-EBS-Volume gehen. Amazon EBS bewahrt den verschlüsselten Volume-Schlüssel für die spätere Verwendung auf, falls der Volume-Schlüssel im Speicher nicht mehr verfügbar ist.

Weitere Informationen zum Verschlüsseln von Amazon-EBS-Volumes mit KMS-Schlüsseln finden Sie unter Wie Amazon Elastic Block Store AWS KMS nutzt im AWS Key Management ServiceEntwicklerhandbuch und Amazon-EBS-Verschlüsselung im Amazon-EC2-Benutzerhandbuch für Linux-Instances und Amazon-EC2-Benutzerhandbuch für Windows-Instances.

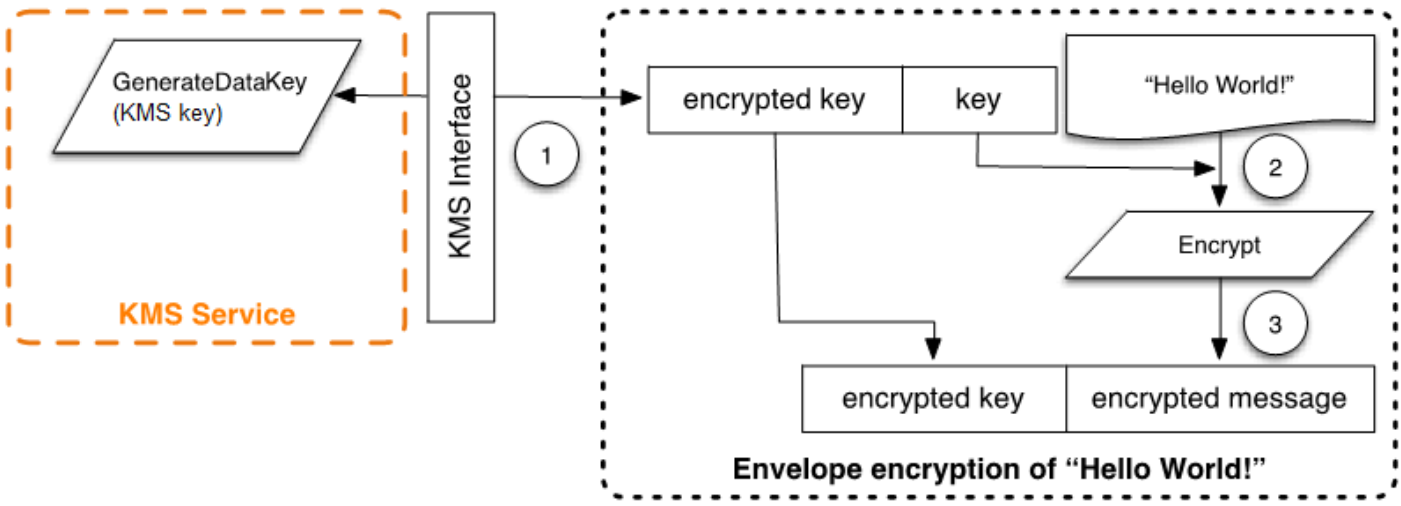
Clientseitige Verschlüsselung

Die AWS Encryption SDK enthält einen API-Vorgang zum Durchführen der Envelope-Verschlüsselung mit einem KMS-Schlüssel. Vollständige Empfehlungen und Details zur Verwendung finden Sie in der zugehörigen Dokumentation. Client-Anwendungen können das AWS Encryption SDK verwenden, um eine Envelope-Verschlüsselung mit AWS KMS durchzuführen.

```
// Instantiate the SDK
final AwsCrypto crypto = new AwsCrypto();
// Set up the KmsMasterKeyProvider backed by the default credentials
final KmsMasterKeyProvider prov = new KmsMasterKeyProvider(keyId);
// Do the encryption
final byte[] ciphertext = crypto.encryptData(prov, message);
```

Die Client-Anwendung kann die folgenden Schritte ausführen:

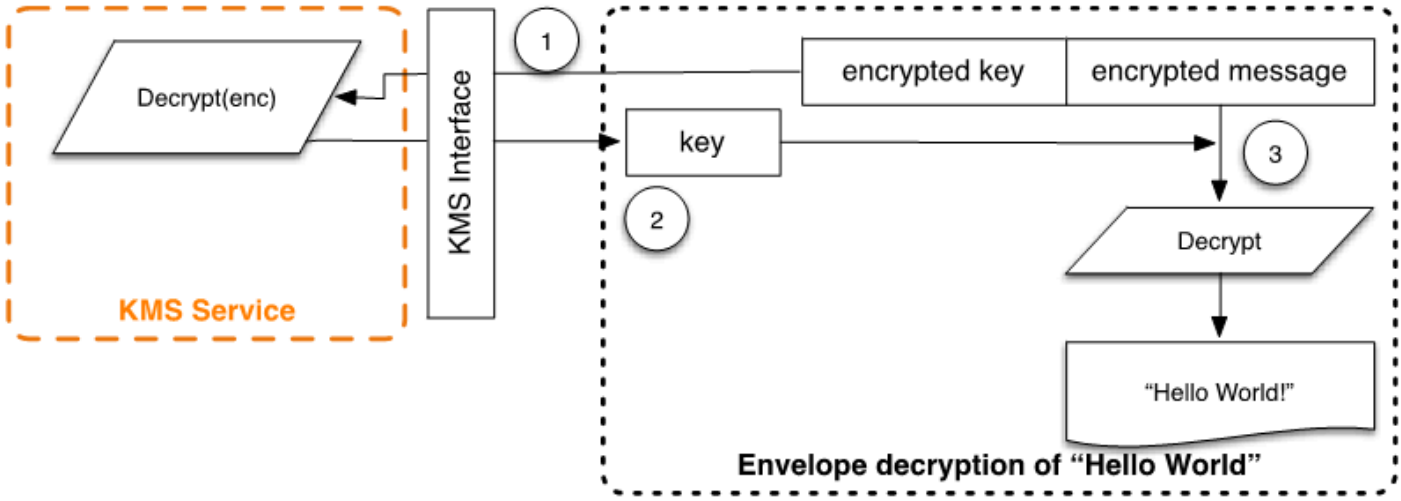
1. Eine Anforderung wird unter einem KMS-Schlüssel für einen neuen Datenschlüssel gestellt. Gibt einen verschlüsselten Datenschlüssel und eine Klartextversion des Datenschlüssels zurück.
2. Innerhalb des AWS Encryption SDK wird der Klartext-Datenschlüssel verwendet, um die Nachricht zu verschlüsseln. Anschließend wird der Datenschlüssel in Klartext aus dem Arbeitsspeicher gelöscht.
3. Der verschlüsselte Datenschlüssel und die verschlüsselte Nachricht werden zu einem einzelnen Chiffretext-Byte-Array kombiniert.



Die Envelope-verschlüsselte Nachricht kann unter Verwendung der Entschlüsselungsfunktionalität entschlüsselt werden, um die ursprünglich verschlüsselte Nachricht zu erhalten.

```
final AwsCrypto crypto = new AwsCrypto();
final KmsMasterKeyProvider prov = new KmsMasterKeyProvider(keyId);
// Decrypt the data
final CryptoResult<byte[], KmsMasterKey> res = crypto.decryptData(prov, ciphertext);
// We need to check the KMS key to ensure that the
// assumed key was used
if (!res.getMasterKeyIds().get(0).equals(keyId)) {
     throw new IllegalStateException("Wrong key id!");
}
byte[] plaintext = res.getResult();
```

1. Das AWS Encryption SDK parst die umschlagverschlüsselte Nachricht, um den verschlüsselten Datenschlüssel zu erhalten und stellt eine Anfrage an AWS KMS, um den Datenschlüssel zu entschlüsseln.
2. AWS Encryption SDK empfängt den Klartext-Datenschlüssel von AWS KMS.
3. Der Datenschlüssel wird dann verwendet, um die Nachricht zu entschlüsseln und den ursprünglichen Klartext zurückzugeben.

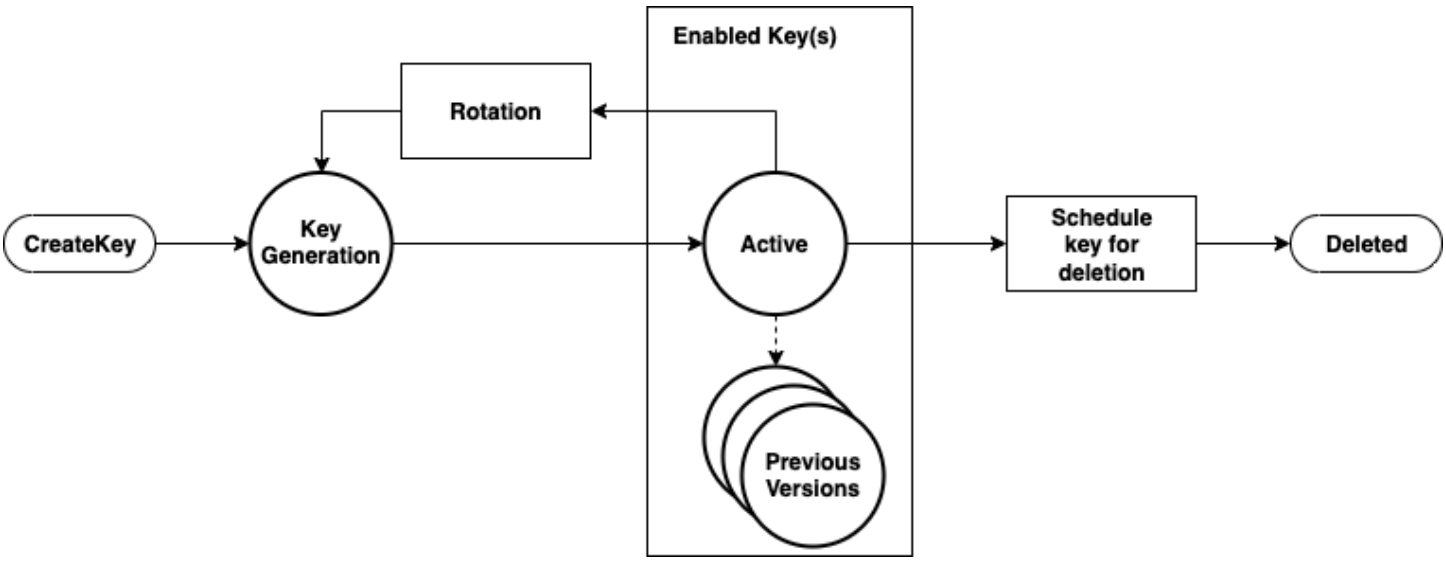

Decrypt(enc)
KMS Interface
KMS Service
1
encrypted key
encrypted message
key
2
3
Decrypt
"Hello World!"
Envelope decryption of "Hello World"

Arbeiten mit AWS KMS keys

Ein AWS KMS key bezieht sich auf einen logischen Schlüssel, der sich auf einen oder mehrere Hardware-Sicherheitsmodul-(HSM)-Backing-Keys (HBKs) beziehen kann. In diesem Thema wird erläutert, wie Sie einen KMS-Schlüssel erstellen, Schlüsselmaterial importieren und KMS-Schlüssel aktivieren, deaktivieren, rotieren und löschen.

> **Note**
>
> AWS KMS ersetzt den Begriff Kundenhauptschlüssel (CMK) durch AWS KMS key und KMS-Schlüssel. Das Konzept hat sich nicht geändert. Um abwärtsinkompatible Änderungen zu vermeiden, werden von AWS KMS einige Varianten dieses Begriffs beibehalten.

In diesem Kapitel wird der Lebenszyklus eines KMS-Schlüssels von der Erstellung bis zur Löschung behandelt, wie in der folgenden Abbildung dargestellt.



Themen

- Aufrufen von CreateKey
- Importieren von Schlüsselmaterial
- Aktivieren und Deaktivieren von Schlüsseln
- Löschen von Schlüsseln
- Drehbares Schlüsselmaterial

Aufrufen von CreateKey

Als Ergebnis eines Aufrufs des `CreateKey`-API-Aufrufs wird ein AWS KMS key generiert.

Das Folgende ist eine Teilmenge der `CreateKey` -Anforderungssyntax.

```
{
  "Description": "string",
  "KeySpec": "string",
  "KeyUsage": "string",
  "Origin": "string";
  "Policy": "string"
}
```

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

Beschreibung

(Optional) Beschreibung des Schlüssels. Wir empfehlen Ihnen, eine Beschreibung auszuwählen, die Ihnen bei der Entscheidung hilft, ob der Schlüssel für eine Aufgabe geeignet ist.

KeySpec

Gibt den Typ des zu erstellenden KMS-Schlüssels an. Der Standardwert, SYMMETRIC_DEFAULT, erstellt einen KMS-Schlüssel mit symmetrischer Verschlüsselung. Dieser Parameter ist für symmetrische Verschlüsselungsschlüssel optional und für alle anderen Schlüsselspezifikationen erforderlich.

KeyUsage

Gibt die Verwendung des Schlüssels an. Gültige Werte sind `ENCRYPT_DECRYPT`, `SIGN_VERIFY` oder `GENERATE_VERIFY_MAC`. Der Standardwert ist `ENCRYPT_DECRYPT`. Dieser Parameter ist für symmetrische Verschlüsselungsschlüssel optional und für alle anderen Schlüsselspezifikationen erforderlich.

Ursprung

(Optional) Gibt die Quelle des Schlüsselmaterials für den KMS-Schlüssel an. Der Standardwert ist `AWS_KMS`, was darauf hindeutet, dass AWS KMS das Schlüsselmaterial für den KMS-Schlüssel generiert und verwaltet. Andere gültige Werte umfassen `EXTERNAL`, der einen KMS-Schlüssel darstellt, der ohne Schlüsselmaterial für importiertes Schlüsselmaterial erstellt wird,

und `AWS_CLOUDHSM`, das einen KMS-Schlüssel in einem benutzerdefinierten Schlüsselspeicher erstellt, der von einem AWS CloudHSM-Cluster unterstützt wird, den Sie steuern.

Richtlinie

Optional: Richtlinie zum Anhängen an den Schlüssel. Wenn die Richtlinie weggelassen wird, wird der Schlüssel mit der Standardrichtlinie (folgend) erstellt, die dem Stamm-Konto und IAM-Prinzipalen mit AWS KMS-Berechtigungen die Verwaltung ermöglicht.

Ausführliche Informationen zur Richtlinie finden Sie unter Schlüsselrichtlinien in AWS KMS und Schlüsselrichtlinien im AWS Key Management Service-Entwicklerhandbuch.

Die `CreateKey`-Anfrage gibt eine Antwort zurück, die einen Schlüssel-ARN enthält.

```
arn:<partition>:kms:<region>:<account-id>:key/<key-id>
```

Wenn der `Origin` gleich `AWS_KMS` ist, wird nach der Erstellung des ARN eine Anfrage an ein AWS KMS-HSM über eine authentifizierte Sitzung gestellt, um einen Hardware Sicherheitsmodul (HSM)-Unterstützungsschlüssel (HBK) bereitzustellen. Der HBK ist ein 256-Bit-Schlüssel, der dieser Schlüssel-ID des KMS-Schlüssels zugeordnet ist. Er kann nur auf einem HSM generiert werden und ist so konzipiert, dass es niemals im Klartext außerhalb der HSM-Grenze exportiert wird. Die HBK wird unter dem aktuellen Domänenschlüssel DK_0 verschlüsselt. Diese verschlüsselten HBKs werden als Encrypted Key Token (EKTs) bezeichnet. Obwohl die HSMs so konfiguriert werden können, dass sie eine Vielzahl von Schlüsselumhüllungsmethoden verwenden, wird bei der aktuellen Implementierung AES-256 im Galois Counter Mode (GCM) verwendet, ein authentifiziertes Verschlüsselungsschema. Dieser authentifizierte Verschlüsselungsmodus ermöglicht es uns, einige im Klartext exportierte Schlüsseltoken-Metadaten zu schützen.

Dies wird stilistisch dargestellt als:

$$EKT = Encrypt(DK_0, HBK)$$

Für Ihre KMS-Schlüssel und die nachfolgenden HBKs werden zwei grundlegende Formen des Schutzes bereitgestellt: Autorisierungsrichtlinien, die für Ihre KMS-Schlüssel festgelegt sind und der kryptografische Schutz für Ihre zugeordneten HBKs. Die restlichen Abschnitte beschreiben den kryptografischen Schutz und die Sicherheit der Verwaltungsfunktionen in AWS KMS.

Zusätzlich zum ARN können Sie einen benutzerfreundlichen Namen erstellen und diesen mit dem KMS-Schlüssel verknüpfen, indem Sie einen Alias für den Schlüssel erstellen. Sobald ein Alias

mit einem KMS-Schlüssel verknüpft wurde, kann der Alias zur Identifizierung des KMS-Schlüssels bei kryptografischen Operationen verwendet werden. Ausführliche Informationen finden Sie unter Verwenden von Aliasen im AWS Key Management Service-Entwicklerhandbuch.

Die Verwendung von KMS-Schlüsseln ist von mehreren Autorisierungsebenen umgeben. AWS KMS aktiviert separate Autorisierungsrichtlinien zwischen dem verschlüsselten Inhalt und dem KMS-Schlüssel. Beispielsweise erbt ein AWS KMS-Envelope-verschlüsseltes Amazon-Simple-Storage-Service-(Amazon-S3)-Objekt die Richtlinie auf dem Amazon-S3-Bucket. Der Zugriff auf den erforderlichen Verschlüsselungsschlüssel wird jedoch durch die Zugriffsrichtlinie für den KMS-Schlüssel bestimmt. Weitere Informationen zur Autorisierung von KMS-Schlüsseln finden Sie unter Authentifizierung und Zugriffssteuerung für AWS KMS im AWS Key Management Service-Entwicklerhandbuch.

Importieren von Schlüsselmaterial

AWS KMS bietet einen Mechanismus zum Importieren des kryptografischen Materials, das für einen HBK verwendet wird. Wie in beschriebenAufrufen von CreateKey, wird bei Verwendung des CreateKey Befehls mit auf `Origin` `EXTERNAL`gesetztem ein logischer KMS-Schlüssel erstellt, der keinen zugrunde liegenden HBK enthält. Das kryptografische Material muss mit dem `ImportKeyMaterial`-API-Aufruf importiert werden. Mit dieser Funktion können Sie die Schlüsselerstellung und Beständigkeit des kryptografischen Materials steuern. Wenn Sie diese Funktion verwenden, empfehlen wir Ihnen, bei der Handhabung und Beständigkeit dieser Tasten in Ihrer Umgebung besondere Vorsicht walten zu lassen. Vollständige Details und Empfehlungen zum Importieren von Schlüsselmaterial finden Sie unter Importieren von Schlüsselmaterial im AWS Key Management Service-Entwicklerhandbuch.

Aufrufen von ImportKeyMaterial

Die `ImportKeyMaterial`-Anfrage importiert das notwendige kryptographische Material für den HBK. Das kryptografische Material muss ein symmetrischer 256-Bit-Schlüssel sein. Es muss mit dem in `WrappingAlgorithm` angegebenen Algorithmus unter dem zurückgegebenen öffentlichen Schlüssel einer kürzlichen `GetParametersForImport`-Anfrage verschlüsselt werden.

Eine `ImportKeyMaterial`-Anfrage übernimmt die folgenden Argumente.

```
{
  "EncryptedKeyMaterial": blob,
  "ExpirationModel": "string",
  "ImportToken": blob,
```

```
  "KeyId": "string",
  "ValidTo": number
}
```

EncryptedKeyMaterial

Das importierte Schlüsselmaterial, das mit dem öffentlichen Schlüssel verschlüsselt wurde, wurde in einer `GetParametersForImport`-Anfrage unter Verwendung des in dieser Anfrage angegebenen Wrapping-Algorithmus, zurückgegeben.

ExpirationModel

Gibt an, ob das Schlüsselmaterial abläuft. Wenn dieser Wert `KEY_MATERIAL_EXPIRES`, die `ValidTo`-Parameter muss ein Ablaufdatum enthalten. Wenn dieser Wert `KEY_MATERIAL_DOES_NOT_EXPIRE` ist, schließen Sie den `ValidTo`-Parameter nicht ein. Die gültigen Werte sind `"KEY_MATERIAL_EXPIRES"` und `"KEY_MATERIAL_DOES_NOT_EXPIRE"`.

ImportToken

Das Import-Token, das von derselben `GetParametersForImport`-Anforderung zurückgegeben wird, die den öffentlichen Schlüssel bereitgestellt hat.

KeyId

Der KMS-Schlüssel, der mit dem importierten Schlüsselmaterial verknüpft werden soll. Das `Origin` des KMS-Schlüssels muss `EXTERNAL` sein.

Sie können dasselbe importierte Schlüsselmaterial löschen und erneut in den angegebenen KMS-Schlüssel importieren, aber Sie können den KMS-Schlüssel nicht mit anderem Schlüsselmaterial importieren oder verknüpfen.

ValidTo

(Optional) Die Zeit, zu der das importierte Schlüsselmaterial abläuft. Wenn das Schlüsselmaterial abgelaufen ist, löscht AWS KMS das Schlüsselmaterial und der KMS-Schlüssel kann nicht mehr verwendet werden. Dieser Parameter ist erforderlich, wenn der Wert von `ExpirationModel` gleich `KEY_MATERIAL_EXPIRES` ist. Andernfalls ist es ungültig.

Wenn die Anfrage erfolgreich war, kann der KMS-Schlüssel innerhalb von AWS KMS bis zum angegebenen Ablaufdatum verwendet werden, falls eines angegeben wurde. Nachdem das importierte Schlüsselmaterial abgelaufen ist, wird das EKT aus der AWS KMS-Speicherebene entfernt.

Aktivieren und Deaktivieren von Schlüsseln

Das Deaktivieren eines KMS-Schlüssels verhindert, dass der Schlüssel in kryptografischen Operationen verwendet wird. Es setzt die Möglichkeit aus, alle HBKs zu verwenden, die dem KMS-Schlüssel zugeordnet sind. Das Aktivieren stellt die Verwendung der HBKs und des KMS-Schlüssels wieder her. Aktivieren und Deaktivieren sind einfache Anforderungen, die nur die Schlüssel-ID oder den Schlüssel-ARN des KMS-Schlüssels benötigen.

Löschen von Schlüsseln

Autorisierte Benutzer können die `ScheduleKeyDeletion`-API verwenden, um die Löschung eines KMS-Schlüssels und aller zugehörigen HBKs zu planen. Dies ist eine von Natur aus destruktive Operation, und Sie sollten beim Löschen von Schlüsseln aus AWS KMS vorsichtig sein. AWS KMS erzwingt beim Löschen von KMS-Schlüsseln eine minimale Wartezeit von sieben Tagen. Während der Wartezeit wird der Schlüssel in einen deaktivierten Zustand mit dem Schlüsselstatus Ausstehende Löschung versetzt. Alle Aufrufe zur Verwendung des Schlüssels für kryptografische Operationen schlagen fehl. ScheduleKeyDeletion verwendet die folgenden Argumente.

```
{
  "KeyId": "string",
  "PendingWindowInDays": number
}
```

KeyId

Der eindeutige Bezeichner für den zu löschenden KMS-Schlüssel. Um diesen Wert anzugeben, verwenden Sie die eindeutige Schlüssel-ID oder den Schlüssel-ARN des KMS-Schlüssels.

PendingWindowInDays

(fakultativ) Die Wartezeit in Anzahl der Tagen. Dieser Wert ist optional. Der Bereich liegt zwischen 7 und 30 Tagen und der Standardwert beträgt 30 Tage. Nach Ablauf der Wartezeit löscht AWS KMS den KMS-Schlüssel und alle zugehörigen HBKs.

Drehbares Schlüsselmaterial

Autorisierte Benutzer können die automatische jährliche Rotation ihrer vom Kunden verwalteten KMS-Schlüssel aktivieren. Von AWS verwaltete Schlüssel werden immer jedes Jahr rotiert.

Wenn ein KMS-Schlüssel rotiert wird, wird eine neue HBK erstellt und als die aktuelle Version des Schlüsselmaterials für alle neuen Verschlüsselungsanfragen markiert. Alle früheren Versionen des HBK bleiben für die Entschlüsselung von Geheimtexten, die mit dieser HBK-Version verschlüsselt wurden, unbegrenzt verfügbar. Da AWS KMS keinen unter einem KMS-Schlüssel verschlüsselten Geheimtext speichert, benötigen Chiffretexte, die unter einer älteren, rotierten HBK verschlüsselt wurden, diese HBK zur Entschlüsselung. Sie können die `ReEncrypt`-API zum erneuten Verschlüsseln eines beliebigen Geheimtextes unter der neuen HBK für den KMS-Schlüssel oder unter einem anderen KMS-Schlüssel verwenden, ohne den Klartext offenzulegen.

Weitere Informationen zum Aktivieren und Deaktivieren der Schlüsselrotation finden Sie unter Rotieren von AWS-KMS-Schlüsseln im AWS Key Management Service-Entwicklerhandbuch.

Kundendatenoperationen

Nachdem Sie einen KMS-Schlüssel eingerichtet haben, kann er zum Ausführen von kryptografischen Operationen verwendet werden. Immer wenn Daten unter einem KMS-Schlüssel verschlüsselt werden, ist das resultierende Objekt ein Kunden-Verschlüsselungstext. Der Verschlüsselungstext enthält zwei Abschnitte: einen unverschlüsselten Header (oder Klartext), der durch das authentifizierte Verschlüsselungsschema als zusätzliche authentifizierte Daten geschützt ist und einen verschlüsselten Teil. Der Klartextbereich enthält den HBK-Bezeichner (HBKID). Diese beiden unveränderlichen Felder des Verschlüsselungstextwerts helfen sicherzustellen, dass AWS KMS das Objekt in Zukunft entschlüsseln kann.

Themen

Generieren eines Datenschlüssels

Autorisierte Benutzer können die GenerateDataKey -API (und zugehörige APIs) verwenden, um einen bestimmten Datenschlüsseltyp oder einen zufälligen Schlüssel beliebiger Länge anzufordern. Dieses Thema bietet eine vereinfachte Ansicht dieser API-Bedienung an. Weitere Informationen finden Sie in den GenerateDataKey APIs in der APIAWS Key Management Service-Referenz zu .

- GenerateDataKey
- GenerateDataKeyWithoutPlaintext
- GenerateDataKeyPair
- GenerateDataKeyPairWithoutPlaintext

Das Folgende ist die `GenerateDataKey`-Anforderungssyntax.

```
{
 “EncryptionContext”: {“string” : “string”},
 “GrantTokens”: [“string”],
 “KeyId”: “string”,
```

```
 “NumberOfBytes”: “number”
}
```

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

KeyId

Schlüssel-ID des Schlüssels, der zum Verschlüsseln des Datenschlüssels verwendet wird. Dieser Wert muss einen KMS-Schlüssel mit symmetrischer Verschlüsselung identifizieren.

Dieser Parameter muss angegeben werden.

NumberOfBytes

Eine Ganzzahl, die die Anzahl der zu generierenden Bytes enthält. Dieser Parameter muss angegeben werden.

Der Anrufer muss entweder `KeySpec` oder `NumberOfBytes` bereitstellen, aber nicht beides.

EncryptionContext

(Optional) Name-Wert-Paar, das zusätzliche Daten zur Authentifizierung während der Verschlüsselungs- und Entschlüsselungsprozesse enthält, die den Schlüssel verwenden.

GrantTokens

(Optional) Eine Liste von Erteilungstoken, die Erteilungen darstellen, die Berechtigungen zum Generieren oder Verwenden eines Schlüssels bereitstellen. Weitere Informationen zu Gewährungen und Gewährungstoken finden Sie unter Authentifizierung und Zugriffssteuerung für AWS KMS im AWS Key Management Service-Entwicklerhandbuch.

AWS KMS erfasst nach der Authentifizierung des Befehls den aktuellen aktiven EKT, der mit dem KMS-Schlüssel verbunden ist. Es leitet den EKT zusammen mit Ihrer bereitgestellten Anfrage und jeglichem Verschlüsselungskontext über eine geschützte Sitzung zwischen dem AWS KMS-Host und einem HSM in der Domäne an ein HSM weiter.

Das HSM macht folgendes:

1. Erzeugt das angeforderte geheime Material und hält es im flüchtigen Speicher.
2. Entschlüsselt den EKT, der der Schlüssel-ID des KMS-Schlüssels entspricht, der in der Anforderung zum Abrufen des aktiven HBK = Decrypt(DK_i, EKT) definiert ist.
3. Generiert eine Zufallszahl N.

4. Generiert einen 256-Bit-AES-GCM-abgeleiteten Verschlüsselungsschlüssel K aus HBK und N.
5. Verschlüsselt das geheime Material ciphertext = Encrypt(K, context, secret).

`GenerateDataKey` gibt das geheime Klartextmaterial und den Geheimtext über den sicheren Kanal zwischen dem AWS KMS-Host und dem HSM an Sie zurück. AWS KMS sendet es dann über die TLS-Sitzung an Sie. AWS KMS behält weder den Klartext noch den Geheimtext bei. Ohne den Verschlüsselungstext, den Verschlüsselungskontext und die Berechtigung zur Verwendung des KMS-Schlüssels kann das zugrunde liegende Geheimnis nicht zurückgegeben werden.

Das Folgende ist die Antwortsyntax.

```
{
 "CiphertextBlob": "blob",
 "KeyId": "string",
 "Plaintext": "blob"
}
```

Die Verwaltung von Datenschlüsseln bleibt Ihnen als Anwendungsentwickler überlassen. Als bewährte Methode zur clientseitigen Verschlüsselung mit AWS KMS-Datenschlüsseln (aber keinen Datenschlüsselpaaren) können Sie die AWS Encryption SDK verwenden.

Datenschlüssel können mit jeder Frequenz gedreht werden. Darüber hinaus kann der Datenschlüssel mithilfe der `ReEncrypt`-API-Operation unter einem anderen KMS-Schlüssel oder einem rotierten KMS-Schlüssel neu verschlüsselt werden. Weitere Informationen finden Sie unter ReEncrypt in der APIAWS Key Management Service-Referenz zu .

Encrypt

Eine grundlegende Funktion von AWS KMS besteht darin, ein Objekt unter einem KMS-Schlüssel zu verschlüsseln. AWS KMS bietet standardmäßig kryptografische Operationen mit geringer Latenz auf HSMs. Somit ist die Menge an Klartext, die in einem direkten Aufruf der Verschlüsselungs-Funktion verschlüsselt werden kann, auf 4 KB begrenzt. AWS Encryption SDK kann verwendet werden, um größere Nachrichten zu verschlüsseln. AWS KMS erwirbt nach der Authentifizierung des Befehls den aktuellen aktiven EKT, der zum KMS-Schlüssel gehört. Es leitet das EKT zusammen mit dem Klartext und dem Verschlüsselungskontext an jedes verfügbare HSM in der Region weiter. Diese werden über eine authentifizierte Sitzung zwischen dem AWS KMS-Host und einem HSM in der Domäne gesendet.

Das HSM führt Folgendes aus:

1. Entschlüsselt den EKT, um HBK = Decrypt(DK_i, EKT) . zu erhalten.
2. Generiert eine Zufallszahl N.
3. Leitet einen 256-Bit-AES-GCM-abgeleiteten Verschlüsselungsschlüssel K von HBK und N ab.
4. Verschlüsselt den Klartext ciphertext = Encrypt(K, context, plaintext).

Der Verschlüsselungstextwert wird an Sie zurückgegeben, und weder die Klartextdaten noch der Verschlüsselungstext werden nirgendwo in der AWS-Infrastruktur gespeichert. Ohne den Verschlüsselungstext, den Verschlüsselungskontext und die Berechtigung zur Verwendung des KMS-Schlüssels kann der zugrunde liegende Klartext nicht zurückgegeben werden.

Decrypt

Ein Aufruf von AWS KMS zum Entschlüsseln eines Verschlüsselungstextwerts akzeptiert einen verschlüsselten Verschlüsselungstextwert und einen Verschlüsselungskontext. AWS KMS authentifiziert den Anruf mithilfe von AWS-Signaturversion-4-signierten Anforderungen und extrahiert die HBKID für den Verpackungsschlüssel aus dem Verschlüsselungstext. Die HBKID wird verwendet, um den EKT zu erhalten, der zum Entschlüsseln des Verschlüsselungstexts, der Schlüssel-ID und der Richtlinie für die Schlüssel-ID erforderlich ist. Die Anforderung wird basierend auf der Schlüsselrichtlinie, möglicherweise vorhandenen Berechtigungen und allen zugehörigen IAM-Richtlinien, die auf die Schlüssel-ID verweisen, autorisiert. Die `Decrypt`-Funktion ist analog zur Verschlüsselungsfunktion.

Das Folgende ist die `Decrypt`-Anforderungssyntax.

```
{
 "CiphertextBlob": "blob",
 "EncryptionContext": { "string" : "string" }
 "GrantTokens": ["string"]
}
```

Im Folgenden sind die Anforderungsparameter aufgeführt.

CiphertextBlob

Verschlüsselter Text einschließlich Metadaten.

EncryptionContext

(Optional) Der Verschlüsselungskontext. Wenn dies in der `Encrypt`-Funktion angegeben wurde, muss es hier angegeben werden, sonst schlägt die Entschlüsselung fehl. Weitere Informationen finden Sie unter Verschlüsselungskontext im AWS Key Management Service-Entwicklerhandbuch.

GrantTokens

(Optional) Eine Liste von Erteilungstoken, die Erteilungen darstellen, die Berechtigungen zum Durchführen der Entschlüsselung bereitstellen.

Der Verschlüsselungstext und das EKT werden zusammen mit dem Verschlüsselungskontext über eine authentifizierte Sitzung an ein HSM zur Entschlüsselung gesendet.

Das HSM führt Folgendes aus:

1. Entschlüsselt den EKT, um HBK = Decrypt(DK_i, EKT) zu erhalten.
2. Extrahiert die Nummer N aus der Verschlüsselungstextstruktur.
3. Regeneriert einen 256-Bit-AES-GCM-abgeleiteten Verschlüsselungsschlüssel K aus HBK und N.
4. Entschlüsselt den Verschlüsselungstext, um plaintext = Decrypt(K, context, ciphertext) zu erhalten.

Die resultierende Schlüssel-ID und Klartext werden über die sichere Sitzung an den AWS KMS-Host und dann über eine TLS-Verbindung an die aufrufende Kundenanwendung zurückgesendet.

Das Folgende ist die Antwortsyntax.

```
{
    "KeyId": "string",
    "Plaintext": blob
}
```

Wenn die aufrufende Anwendung die Authentizität des Klartexts sicherstellen möchte, muss sie überprüfen, ob die zurückgegebene Schlüssel-ID der erwarteten entspricht.

Erneutes Verschlüsseln eines verschlüsselten Objekts

Ein vorhandener verschlüsselter Kundentext, der unter einem KMS-Schlüssel verschlüsselt wurde, kann über einen reencrypt-Befehl zum erneuten Verschlüsseln mit einem anderen KMS-Schlüssel

neu verschlüsselt werden. Reencrypt verschlüsselt Daten auf der Serverseite mit einem neuen KMS-Schlüssel, ohne den Klartext des Schlüssels auf der Clientseite offenzulegen. Die Daten werden zuerst entschlüsselt und dann verschlüsselt.

Das Folgende ist die -Anforderungssyntax.

```
{
 "CiphertextBlob": "blob",
 "DestinationEncryptionContext": { "string" : "string" },
 "DestinationKeyId": "string",
 "GrantTokens": ["string"],
       "SourceKeyId": "string",
 "SourceEncryptionContext": { "string" : "string"}
}
```

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

CiphertextBlob

Verschlüsselungstext der Daten, die neu verschlüsselt werden sollen.

DestinationEncryptionContext

(Optional) Verschlüsselungskontext, der verwendet werden soll, wenn die Daten neu verschlüsselt werden.

DestinationKeyId

Schlüsselkennung des Schlüssels, der zum erneuten Verschlüsseln der Daten verwendet wird.

GrantTokens

(Optional) Eine Liste von Erteilungstoken, die Erteilungen darstellen, die Berechtigungen zum Durchführen der Entschlüsselung bereitstellen.

SourceKeyId

(Optional) Schlüssel-ID des Schlüssels, der zum Entschlüsseln der Daten verwendet wird.

SourceEncryptionContext

(Optional) Verschlüsselungskontext, der zum Verschlüsseln und Entschlüsseln der Daten verwendet wird, die im `CiphertextBlob`-Parameter angegeben sind.

Der Prozess kombiniert die Entschlüsselungs- und Verschlüsselungsoperationen der vorherigen Beschreibungen: Der Kunden-Verschlüsselungstext wird unter dem anfänglichen HBK, auf den der Kunden-Verschlüsselungstext verweist, zum aktuellen HBK unter dem beabsichtigten KMS-Schlüssel entschlüsselt. Wenn die in diesem Befehl verwendeten KMS-Schlüssel identisch sind, verschiebt dieser Befehl den Kunden-Verschlüsselungstext von einer alten Version eines HBK in die neueste Version eines HBK.

Das Folgende ist die Antwortsyntax.

```
{
    "CiphertextBlob": blob,
    "DestinationEncryptionAlgorithm": "string",
    "KeyId": "string",
    "SourceEncryptionAlgorithm": "string",
    "SourceKeyId": "string"
}
```

Wenn die aufrufende Anwendung die Authentizität des zugrunde liegenden Klartexts sicherstellen möchte, muss sie überprüfen, ob die SourceKeyId zurückgegebene die erwartete ist.

AWS KMS interne Operationen

AWS KMS-Interna sind erforderlich, um HSMs für einen global verteilten Schlüsselverwaltungsservice zu skalieren und zu sichern.

Themen

Domänen und Domänenstatus

Eine kooperative Sammlung von vertrauenswürdigen internen AWS KMS-Entitäten innerhalb eines AWS-Region wird als Domäne bezeichnet. Eine Domäne umfasst einen Satz vertrauenswürdiger Entitäten, einen Satz von Regeln und einen Satz geheimer Schlüssel, die als Domänenschlüssel bezeichnet werden. Die Domänenschlüssel werden von HSMs freigegeben, die Mitglieder der Domäne sind. Ein Domänenstatus besteht aus den folgenden Feldern.

Name

Ein Domänenname zur Identifizierung dieser Domäne.

Mitglieder

Eine Liste der HSMs, die Mitglieder der Domäne sind, einschließlich ihres öffentlichen Signaturschlüssels und der Schlüssel zur öffentlichen Vereinbarung.

Operatoren

Eine Liste von Entitäten, öffentlichen Signaturschlüsseln und einer Rolle (AWS KMS-Operator oder Servicehost), die die Betreiber dieses Services darstellt.

Regeln

Eine Liste der Quorumregeln für jeden Befehl, der erfüllt sein muss, um einen Befehl auf dem HSM auszuführen.

Domänenschlüssel

Eine Liste der Domänenschlüssel (symmetrische Schlüssel), die derzeit in der Domäne verwendet werden.

Der vollständige Domänenstatus ist nur im HSM verfügbar. Der Domänenstatus wird zwischen HSM-Domänenmitgliedern als exportiertes Domänen-Token synchronisiert.

Domänenschlüssel

Alle HSMs in einer Domäne teilen sich einen Satz von Domänenschlüsseln $\{DK_r\}$. Diese Schlüssel werden über eine Exportroutine für Domänenstatus freigegeben. Der exportierte Domänenstatus kann in jedes HSM importiert werden, das Mitglied der Domäne ist.

Der Satz von Domänenschlüsseln $\{DK_r\}$enthält immer einen aktiven Domänenschlüssel und mehrere deaktivierte Domänenschlüssel. Domänenschlüssel werden täglich gedreht, um sicherzustellen, dass AWS die Empfehlung für Schlüsselmanagement - Teil 1 erfüllt. Während der Domänenschlüsseldrehung werden alle vorhandenen KMS-Schlüssel, die unter dem ausgehenden Domänenschlüssel verschlüsselt wurden, unter dem neuen aktiven Domänenschlüssel neu verschlüsselt. Der aktive Domänenschlüssel wird verwendet, um neue EKTs zu verschlüsseln. Die abgelaufenen Domänenschlüssel können nur verwendet werden, um zuvor verschlüsselte EKTs für eine Anzahl von Tagen zu entschlüsseln, die der Anzahl der kürzlich gedrehten Domänenschlüssel entspricht.

Exportierte Domänen-Token

Es besteht eine regelmäßige Notwendigkeit, den Status zwischen Domänenteilnehmern zu synchronisieren. Dies wird durch den Export des Domänenstatus erreicht, wenn eine Änderung an der Domäne vorgenommen wird. Der Domänenstaus wird als exportiertes Domänen-Token exportiert.

Name

Ein Domänenname zur Identifizierung dieser Domäne.

Mitglieder

Eine Liste der HSMs, die Mitglieder der Domäne sind, einschließlich der öffentlichen Schlüssel zur Unterzeichnung und Vereinbarung.

Operatoren

Eine Liste von Entitäten, öffentlichen Signaturschlüsseln und einer Rolle, die die Operatoren dieses Services darstellt.

Regeln

Eine Liste der Quorumregeln für jeden Befehl, der erfüllt sein muss, um einen Befehl auf einem HSM-Domänenmitglied auszuführen.

Verschlüsselte Domänenschlüssel

Envelope-verschlüsselte Domänenschlüssel. Die Domänenschlüssel werden vom unterzeichnenden Mitglied für jedes der oben aufgeführten Mitglieder verschlüsselt und in seinen öffentlichen Vereinbarungsschlüssel Envelope-verschlüsselt.

Signatur

Eine Signatur im Domänenstatus, die von einem HSM erstellt wurde, notwendigerweise ein Mitglied der Domäne, die den Domänenstatus exportiert hat.

Das exportierte Domänen-Token bildet die grundlegende Vertrauensquelle für Entitäten, die innerhalb der Domäne arbeiten.

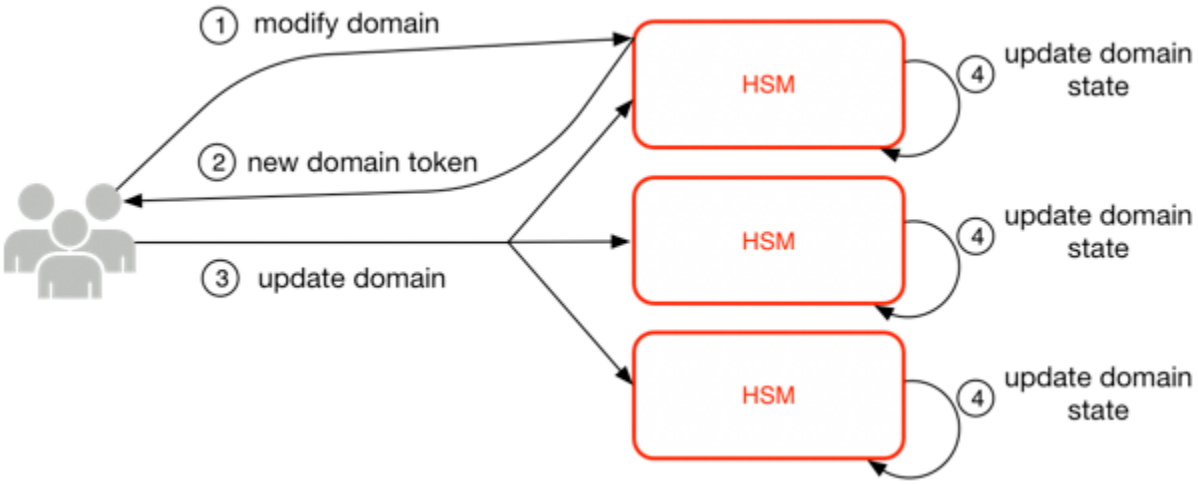
Verwalten von Domänenstatus

Der Domänenstatus wird über quorumauthentifizierte Befehle verwaltet. Zu diesen Änderungen gehören das Ändern der Liste der vertrauenswürdigen Teilnehmer in der Domäne, das Ändern der Quorumregeln zum Ausführen von HSM-Befehlen und das regelmäßige Drehen der Domänenschlüssel. Diese Befehle werden pro Befehl authentifiziert, im Gegensatz zu authentifizierten Sitzungsvorgängen, wie im folgenden Bild gezeigt.

Ein HSM enthält in seinem initialisierten und betriebsbereiten Status einen Satz selbsterzeugter asymmetrischer Identitätsschlüssel, ein Signaturschlüsselpaar und ein Schlüsselpaar zur Schlüsseleinrichtung. Durch einen manuellen Prozess kann ein AWS KMS-Operator eine anfängliche Domäne einrichten, die auf einem ersten HSM in einer Region erstellt werden soll. Diese anfängliche Domäne besteht aus einem vollständigen Domänenstatus, wie zuvor in diesem Thema definiert. Es wird über einen Join-Befehl zu jedem der definierten HSM-Mitglieder in der Domäne installiert.

Nachdem ein HSM einer anfänglichen Domäne beigetreten ist, ist es an die Regeln gebunden, die in dieser Domäne definiert sind. Diese Regeln regeln die Befehle, die Kryptographieschlüssel

des Kunden verwenden oder Änderungen am Host- oder Domänenstatus vornehmen. Die authentifizierten Sitzungs-API-Operationen, die Ihre kryptografischen Schlüssel verwenden, wurden früher definiert.



Das obige Bild zeigt, wie ein Domänenstatus geändert wird. Der Prozess besteht aus vier Schritten:

1. Ein quorumbasierter Befehl wird an ein HSM gesendet, um die Domäne zu ändern.
2. Ein neuer Domänenstatus wird generiert und als neues exportiertes Domänentoken exportiert. Der Status auf dem HSM wird nicht geändert, was bedeutet, dass die Änderung nicht auf dem HSM übernommen wird.
3. Ein zweiter Befehl wird an jede der HSMs im neu exportierten Domänentoken gesendet, um ihren Domänenstatus mit dem neuen Domänentoken zu aktualisieren.
4. Die HSMs, die im neuen exportierten Domänentoken aufgeführt sind, können den Befehl und das Domänentoken authentifizieren. Sie können auch die Domänenschlüssel entpacken, um den Domänenstatus für alle HSMs in der Domäne zu aktualisieren.

HSMs kommunizieren nicht direkt miteinander. Stattdessen fordert ein Quorum von Operatoren eine Änderung des Domänenstatus an, die zu einem neuen exportierten Domänen-Token führt. Ein Servicehostmitglied der Domäne wird verwendet, um den neuen Domänenstatus an jedes HSM in der Domäne zu verteilen.

Das Verlassen und Verbinden einer Domäne erfolgt über die HSM-Verwaltungsfunktionen. Die Änderung des Domänenstatus erfolgt über die Domänenverwaltungsfunktionen.

Verlassen der Domäne

Bewirkt, dass ein HSM eine Domäne verlässt und alle Reste und Schlüssel dieser Domäne aus dem Speicher löscht.

Registrieren der Domäne

Bewirkt, dass ein HSM einer neuen Domäne beitritt oder seinen aktuellen Domänenstatus auf den neuen Domänenstatus aktualisiert. Die vorhandene Domäne wird als Quelle des anfänglichen Regelsatzes verwendet, um diese Nachricht zu authentifizieren.

Domäne erstellen

Bewirkt, dass eine neue Domäne auf einem HSM erstellt wird. Gibt ein erstes Domänentoken zurück, das an Mitglieds-HSMs der Domäne verteilt werden kann.

Operatoren ändern

Fügt Operatoren und deren Rollen in der Domäne hinzu oder entfernt sie aus der Liste der autorisierten Operatoren.

Ändern von Mitgliedern

Fügt ein HSM aus der Liste der autorisierten HSMs in der Domäne hinzu oder entfernt es.

Regeln ändern

Ändert den Satz von Quorumregeln, die zum Ausführen von Befehlen auf einem HSM erforderlich sind.

Drehen der Domänenschlüssel

Bewirkt, dass ein neuer Domänenschlüssel erstellt und als aktiver Domänenschlüssel markiert wird. Dadurch wird der vorhandene aktive Schlüssel in einen deaktivierten Schlüssel verschoben und der älteste deaktivierte Schlüssel aus dem Domänenstatus entfernt.

Interne Kommunikationssicherheit

Befehle zwischen den Service-Hosts oder AWS KMS-Operatoren und den HSMs werden durch zwei in Authentifizierte Sitzungen dargestellte Mechanismen gesichert: eine quorumsignierte Anforderungsmethode und eine authentifizierte Sitzung unter Verwendung eines HSM-Service-Host-Protokolls.

Die quorumsignierten Befehle sind so konzipiert, dass kein einzelner Operator die kritischen Sicherheitsvorkehrungen ändern kann, die von den HSMs bereitgestellt werden. Die Befehle, die über die authentifizierten Sitzungen ausgeführt werden, tragen dazu bei, dass nur autorisierte Service-Operatoren Operationen mit KMS-Schlüsseln ausführen können. Alle kundengebundenen geheimen Informationen werden über die AWS-Infrastruktur gesichert.

Schlüssel-Einrichtung

Um die interne Kommunikation abzusichern, verwendet AWS KMS zwei verschiedene Methoden zur Schlüsseleinrichtung. Die erste ist als C(1, 2, ECC DH) in der Empfehlung für Paarweise Schlüsseleinrichtungsschemata mit diskreter Logarithmus-Kryptographie (Revision 2) definiert. Dieses Schema hat einen Initiator mit einem statischen Signaturschlüssel. Der Initiator generiert und signiert einen ephemeren elliptischen Kurven-Diffie-Hellman-Schlüssel (ECDH), der für einen Empfänger mit einem statischen ECDH-Übereinstimmungsschlüssel bestimmt ist. Diese Methode verwendet einen ephemeren Schlüssel und zwei statische Schlüssel mit ECDH. Das ist die Ableitung der Markierung C (1, 2, ECC DH). Diese Methode wird manchmal als One-Pass-ECDH bezeichnet.

Die zweite Methode zur Schlüsseleinrichtung ist C(2, 2, ECC, DH). In diesem Schema haben beide Parteien einen statischen Signaturschlüssel, und sie generieren, signieren und tauschen einen ephemeren ECDH-Schlüssel aus. Diese Methode verwendet zwei statische Schlüssel und zwei ephemere Schlüssel, die jeweils ECDH verwenden. Das ist die Ableitung der Markierung C (2, 2, ECC DH). Diese Methode wird manchmal als ephemere ECDH oder ECDHE bezeichnet. Alle ECDH-Schlüssel werden auf der Kurve secp384r1 (NIST-P384) generiert.

HSM-Sicherheitsgrenze

Die innere Sicherheitsgrenze von AWS KMS ist das HSM. Das HSM verfügt über eine proprietäre Schnittstelle und keine anderen aktiven physikalischen Schnittstellen im Betriebszustand. Ein operatives HSM wird während der Initialisierung mit den erforderlichen kryptografischen Schlüsseln bereitgestellt, um seine Rolle in der Domäne festzulegen. Sensible kryptografische Materialien des HSM werden nur im flüchtigen Speicher gespeichert und gelöscht, wenn das HSM den Betriebszustand verlässt, einschließlich beabsichtigter oder unbeabsichtigter Abschaltungen oder Rücksetzungen.

Die HSM-API-Operationen werden entweder durch einzelne Befehle oder über eine gegenseitig authentifizierte vertrauliche Sitzung authentifiziert, die von einem Service-Host eingerichtet wird.

Quorumsignierte Befehle

Quorumsignierte Befehle werden von Operatoren an HSMs ausgegeben. In diesem Abschnitt wird beschrieben, wie quorumbasierte Befehle erstellt, signiert und authentifiziert werden. Diese Regeln sind ziemlich einfach. Beispielsweise erfordert der Befehl Foo, dass zwei Mitglieder der Rolle Bar authentifiziert werden. Es gibt drei Schritte bei der Erstellung und Überprüfung eines quorumbasierten Befehls. Der erste Schritt ist die anfängliche Befehlserstellung; die zweite ist die Vorlage an zusätzliche Operatoren zur Unterzeichnung; und die dritte ist die Überprüfung und Ausführung.

Um die Konzepte einzuführen, gehen Sie davon aus, dass es einen authentischen Satz von öffentlichen Schlüsseln und Rollen des Operatoren $\{QOS_s\}$ und eine Reihe von Quorum-Regeln QR = $\{Befehl_i, Regel_{\{i, t\}}\}$ gibt, wobei jede Regel eine Reihe von Rollen und minimale Anzahl N $\{Rolle_t, N_t\}$ ist. Damit ein Befehl die Quorumregel erfüllt, muss die Befehlsdatei von einer in $\{QOS_s\}$ aufgelisteten Gruppe von Operatoren signiert sein, sodass sie eine der für diesen Befehl aufgeführten Regeln erfüllen. Wie bereits erwähnt, werden die Quorumregeln und Operatoren im Domänenstatus und im exportierten Domänen-Token gespeichert.

In der Praxis signiert ein Initialsignierer den Befehl $Sig_1 = Sign(dO_{p1}, Befehl)$. Ein zweiter Operator signiert auch den Befehl $Sig_2 = Sign(dO_{p2}, Befehl)$. Die doppelt signierte Nachricht wird zur Ausführung an ein HSM gesendet. Die HMS führt Folgendes aus:

1. Für jede Signatur extrahiert er den öffentlichen Schlüssel des Unterzeichners aus dem Domänenstatus und überprüft die Signatur des Befehls.
2. Es überprüft, ob die Gruppe von Unterzeichnern eine Regel für den Befehl erfüllt.

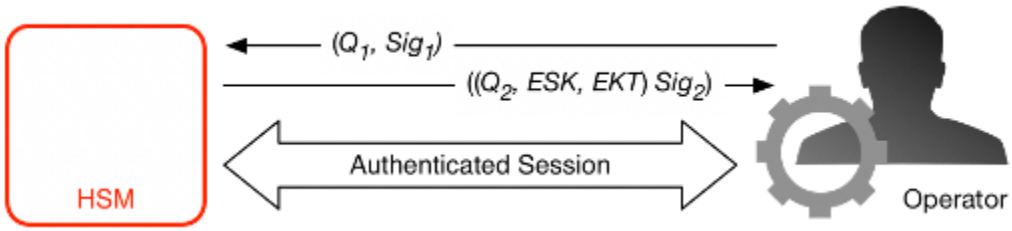
Authentifizierte Sitzungen

Ihre Schlüsseloperationen werden zwischen den nach außen gerichteten AWS KMS-Hosts und den HSMs ausgeführt. Diese Befehle beziehen sich auf die Erstellung und Verwendung von

kryptografischen Schlüsseln und die sichere Zufallszahlengenerierung. Die Befehle werden über einen sitzungs-authentifizierten Kanal zwischen den Services-Hosts und den HSMs ausgeführt. Neben der Notwendigkeit der Authentizität erfordern diese Sitzungen Vertraulichkeit. Befehle, die über diese Sitzungen ausgeführt werden, umfassen die Rückgabe von Klartext-Datenschlüsseln und entschlüsselten Nachrichten, die für Sie bestimmt sind. Um sicherzustellen, dass diese Sitzungen nicht durch man-in-the-middle Angriffe unterlaufen werden können, werden Sitzungen authentifiziert.

Dieses Protokoll führt eine gegenseitig authentifizierte ECDHE-Schlüsselvereinbarung zwischen dem HSM und dem Service-Host durch. Der Austausch wird vom Service-Host initiiert und vom HSM abgeschlossen. Der HSM gibt auch einen durch den ausgehandelten Schlüssel verschlüsselten Sitzungsschlüssel (SK) und einen exportierten Schlüssel-Token zurück, der den Sitzungsschlüssel enthält. Das exportierte Schlüssel-Token enthält einen Gültigkeitszeitraum, nach dem der Service-Host einen Sitzungsschlüssel neu aushandeln muss.

Ein Service-Host ist Mitglied der Domäne und verfügt über ein identitätssignierendes Schlüsselpaar ($DHOS_i$, $QHOS_i$) und eine authentische Kopie der öffentlichen Schlüssel der HSMs. Es verwendet seinen Satz von Identitätssignierungsschlüsseln, um sicher einen Sitzungsschlüssel auszuhandeln, der zwischen dem Service-Host und jedem HSM in der Domäne verwendet werden kann. Den exportierten Schlüssel-Token ist ein Gültigkeitszeitraum zugeordnet, nach dem ein neuer Schlüssel ausgehandelt werden muss.



Der Prozess beginnt damit, dass der Service-Host erkennt, dass er einen Sitzungsschlüssel benötigt, um vertrauliche Kommunikationsflüsse zwischen ihm und einem HSM-Mitglied der Domäne zu senden und zu empfangen.

1. Ein Service-Host generiert ein ephemeres ECDH-Schlüsselpaar (d_1, Q_1) und signiert es mit seinem Identitätsschlüssel Sig1 = Sign(dOS,Q_1).
2. Das HSM überprüft die Signatur auf dem empfangenen öffentlichen Schlüssel mit seinem aktuellen Domänen-Token und erstellt ein ephemeres ECDH-Schlüsselpaar (d_2, Q_2). Es schließt dann den ECDH-Schlüsselaustausch gemäß der Empfehlung für Paarweise Schlüsseleinrichtungsschemata mit diskreter Logarithmus-Kryptographie (überarbeitet) ab, um einen ausgehandelten 256-Bit-AES-GCM-Schlüssel zu bilden. Das HSM generiert einen neuen 256-Bit-AES-GCM-

Sitzungsschlüssel. Es verschlüsselt den Sitzungsschlüssel mit dem ausgehandelten Schlüssel, um den verschlüsselten Sitzungsschlüssel (ESK) zu bilden. Es verschlüsselt auch den Sitzungsschlüssel unter dem Domänenschlüssel als exportiertes Schlüssel-Token EKT. Schließlich signiert es einen Rückgabewert mit seinem Identitätsschlüsselpaar Sig_2 = Sign(dHSK, (Q_2, ESK, EKT)).

3. Der Service-Host überprüft die Signatur für die empfangenen Schlüssel mit seinem aktuellen Domänen-Token. Der Service-Host schließt dann den ECDH-Schlüsselaustausch gemäß der -Empfehlung für paarweise Schlüsseleinrichtungsschemata unter Verwendung von diskreter Logarithmus-Kryptographie (überarbeitet) ab. Es entschlüsselt als nächstes den ESK, um den Sitzungsschlüssel SK zu erhalten.

Während der Gültigkeitsdauer im EKT kann der Service-Host den ausgehandelten Sitzungsschlüssel SK verwenden, um mit einen Envelope-verschlüsselten Befehl an das HSM zu senden. Jeder service-host-initiated Befehl über diese authentifizierte Sitzung enthält den EKT . Der HSM antwortet mit demselben ausgehandelten Sitzungsschlüssel SK.

Replikationsprozess für Schlüssel mit mehreren Regionen

AWS KMS verwendet einen regionenübergreifenden Replikationsmechanismus, um das Schlüsselmaterial eines KMS-Schlüssels von einem HSM in einer AWS-Region zu einem HSM in einer anderen AWS-Region zu kopieren. Damit dieser Mechanismus funktioniert, muss der zu replizierende KMS-Schlüssel ein Multiregions-Schlüssel sein. Wenn Sie einen KMS-Schlüssel von einer Region in eine andere replizieren, können die HSMs in den Regionen nicht direkt kommunizieren, weil sie sich in isolierten Netzwerken befinden. Stattdessen werden die während der regionsübergreifenden Replikation ausgetauschten Nachrichten von einem Proxyservice übermittelt.

Bei der regionsübergreifenden Replikation wird jede von einem AWS KMS-HSM generierte Nachricht kryptografisch mit einem Signierschlüssel für Replikation signiert. Replikationssignierschlüssel (RSKs) sind ECDSA-Schlüssel auf der NIST-P-384-Kurve. Jede Region besitzt mindestens eine RSK, und die öffentliche Komponente jedes RSK wird mit jeder anderen Region in derselben AWS-Partition geteilt.

Der regionsübergreifende Replikationsprozess zum Kopieren von Schlüsselmaterial von Region A nach Region B funktioniert folgendermassen:

1. Das HSM in Region B erzeugt einen flüchtigen ECDH-Schlüssel auf der NIST-P-384-Kurve, Replikationsvereinbarungs-Schlüssel B (RAKB). Die öffentliche Komponente von RAKB wird vom Proxy-Dienst an ein HSM in Region A gesendet.
2. Das HSM in Region A erhält die öffentliche Komponente von RAKB und generiert dann einen weiteren flüchtigen ECDH-Schlüssel auf der NIST-P-384-Kurve, Replikationsvereinbarungs-Schlüssel A (RAKA). Dazu führt das HSM das ECDH-Schlüsselherstellungsschema für RAKA und die öffentliche Komponente von RAKB aus und leitet aus der Ausgabe einen symmetrischen Schlüssel ab, den Replication Wrapping Key (RWK). Die RWK wird verwendet, um das Schlüsselmaterial des multiregionalen KMS-Schlüssels zu verschlüsseln, der repliziert wird.
3. Die öffentliche Komponente von RAKA und das mit der RWK verschlüsselte Schlüsselmaterial werden über den Proxy-Service an das HSM in Region B gesendet.
4. Das HSM in Region B erhält die öffentliche Komponente von RAKA und das Schlüsselmaterial, das mit der RWK verschlüsselt wurde. Das HSM wird von RWK abgeleitet, indem das ECDH-Schlüsselherstellungsschema auf RAKB und der öffentlichen Komponente von RAKA ausgeführt wird.
5. Das HSM in Region B verwendet die RWK, um das Schlüsselmaterial aus Region A zu entschlüsseln.

Beständigkeitsschutz

Zusätzliche Service-Beständigkeit für vom Service generierte Schlüssel wird durch die Verwendung von Offline-HSMs, mehrfacher nichtflüchtiger Speicherung exportierter Domänentoken und redundanter Speicherung verschlüsselter KMS-Schlüssel bereitgestellt. Die Offline-HSMs sind Mitglieder der vorhandenen Domänen. Mit der Ausnahme, dass sie nicht online sind und am regulären Domänenbetrieb teilnehmen, werden die Offline-HSMs im Domänenstatus identisch mit den vorhandenen HSM-Mitgliedern angezeigt.

Das Beständigkeitsdesign soll alle KMS-Schlüssel in einer Region schützen, falls AWS einen großflächigen Verlust der Online-HSMs oder der in unserem primären Speichersystem gespeicherten KMS-Schlüssel erleidet. AWS KMS keys mit importiertem Schlüsselmaterial fallen nicht unter den Beständigkeitsschutz, den andere KMS-Schlüssel bieten. Im Falle eines landesweiten Fehlers in AWS KMS muss importiertes Schlüsselmaterial möglicherweise erneut in einen KMS-Schlüssel importiert werden.

Die Offline-HSMs und die Zugangsdaten werden in Safes in überwachten Sicherheitsräumen an mehreren unabhängigen geografischen Standorten gespeichert. Jeder Safe erfordert mindestens

einen AWS-Sicherheitsbeauftragten und einen AWS KMS-Bediener aus zwei unabhängigen Teams in AWS, um diese Materialien zu erhalten. Die Verwendung dieser Materialien wird durch interne Richtlinien geregelt, die die Anwesenheit eines Quorums von AWS KMS-Bedienern erfordern.

Referenz

Verwenden Sie das folgende Referenzmaterial, um Informationen zu Abkürzungen, Schlüsseln, Mitwirkenden und Quellen in diesem Dokument zu erhalten.

Themen

Abkürzungen

In der folgenden Liste werden Abkürzungen angezeigt, auf die in diesem Dokument verwiesen wird.

AES

: Erweiterter Verschlüsselungsstandard

CDK

: Kunden-Datenschlüssel

DK

: Domänenschlüssel

ECDH

: Elliptische Kurve Diffie-Hellman

ECDHE

: Elliptische Kurve Diffie-Hellman kurzlebig

ECDSA

: Elliptischer-Kurven-Digital-Signatur-Algorithmus

EKT

: Exportiertes Schlüssel-Token

ESK

Verschlüsselter Sitzungsschlüssel

GCM

Galois-Zähler-Modus

HBK

HSM-Unterstützungsschlüssel

HBKID

HSM-Unterstützungsschlüssel-Bezeichner

HSM

Hardwaresicherheitsmodul

RSA

Rivest Shamir und Adleman (kryptologic)

secp384r1

Standards für effiziente Kryptographie Prime 384-Bit Zufallskurve 1

SHA256

Sicherer Hash-Algorithmus mit einer Digest-Länge von 256 Bit

Schlüssel

In der folgenden Liste werden die Schlüssel definiert, auf die in diesem Dokument verwiesen wird.

HBK

HSM-Unterstützungsschlüssel: HSM-Unterstützungsschlüssel sind 256-Bit-Stamm-Schlüssel, von denen spezifische Verwendungsschlüssel abgeleitet werden.

DK

Domänenschlüssel: Ein Domänenschlüssel ist ein 256-Bit-AES-GCM-Schlüssel. Sie wird von allen Mitgliedern einer Domäne gemeinsam genutzt und zum Schutz des HSM-Unterstützungsschlüssel-Materials und der HSM-Service-Host-Sitzungsschlüssel verwendet.

DKEK

Verschlüsselungsschlüssel für Domänenschlüssel: Ein Domänenschlüsselverschlüsselungsschlüssel ist ein AES-256-GCM-Schlüssel, der auf einem Host generiert und zum Verschlüsseln des aktuellen Domänenschlüssels verwendet wird, der den Domänenstatus über die HSM-Hosts hinweg synchronisiert.

(dHAK,QHAK)

HSM-Vereinbarungsschlüsselpaar: Jedes initiierte HSM hat ein lokal generiertes elliptische Kurven-Diffie-Hellman-Vereinbarungsschlüsselpaar auf der Kurve secp384r1 (NIST-P384).

(dE, QE)

Schlüsselpaar für kurzlebige Vereinbarungen: HSM und Diensthosts generieren kurzlebige Vereinbarungsschlüssel. Dies sind elliptische Kurven-Diffie-Hellman–Schlüssel auf der Kurve secp384r1 (NIST-P384). Diese werden in zwei Anwendungsfällen generiert: zum Einrichten eines host-to-host Verschlüsselungsschlüssels für den Transport von Domänenschlüssel-Verschlüsselungsschlüsseln in Domänen-Token und zum Einrichten von HSM-Service-Host-Sitzungsschlüsseln zum Schutz sensibler Kommunikation.

(dHSK, QHSK)

HSM-Signatur-Schlüsselpaar: Jedes initiierte HSM hat ein lokal generiertes elliptische Kurven-Digital-Signatur-Schlüsselpaar auf der Kurve secp384r1 (NIST-P384).

(dOS, QOS)

Operator-Signatur-Schlüsselpaar: Sowohl die Service-Host-Operatoren als auch die AWS KMS-Operatoren verfügen über einen Identitäts-Signaturschlüssel, der verwendet wird, um sich gegenüber anderen Domänenteilnehmern zu authentifizieren.

K

Datenverschlüsselungsschlüssel: Ein 256-Bit-AES-GCM-Schlüssel, der von einem HBK mit NIST SP800-108 KDF im Zählermodus mit HMAC mit SHA256 abgeleitet wurde.

SK

Sitzungsschlüssel: Ein Sitzungsschlüssel wird als Ergebnis eines authentifizierten elliptischen Kurven-Diffie-Hellman-Schlüssels erstellt, der zwischen einem Service-Host-Operator und einem HSM ausgetauscht wird. Der Zweck des Austauschs besteht darin, die Kommunikation zwischen dem Diensthost und den Mitgliedern der Domäne zu sichern.

Mitwirkende

Folgende Personen und Organisationen haben zu diesem Dokument beigetragen:

- Ken Beer, Geschäftsführer - KMS,AWS Kryptografie
- Matthew Campagna, Sicherheitsingenieur, AWS Kryptografie

Bibliographie

Weitere Informationen über die AWS Key Management Service-HSMs, gehen Sie zur NIST-Computersicherheitsressourcencenter Seite zum Validierungsprogramm für Kryptographische Module und suchen Sie nach AWS Key Management Service-HSM.

Amazon Web Services, Allgemeine Referenz (Version 1.0), „AWS-API-Anforderung signieren“http://docs.aws.amazon.com/general/latest/gr/signing_aws_api_requests.htmlaus.

Amazon Web Services, „Was ist AWS Encryption SDK“ http://docs.aws.amazon.com/encryption-sdk/latest/developer-guide/introduction.html.

Federal Information Processing Standards Publications, FIPS PUB 180-4. Secure Hash Standard, August 2012. Verfügbar unter https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf.

Federal Information Processing Standards Publication 197, Ankündigung des Advanced Encryption Standard (AES), November 2001. Verfügbar unter http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

Federal Information Processing Standards Publication 198-1, The Keyed-Hash Message Authentication Code (HMAC), Juli 2008. Verfügbar unter http://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf.

NIST Special Publication 800-52 Revision 2, Richtlinien für die Auswahl, Konfiguration und Verwendung von Transport Layer Security (TLS)-Implementierungen, August 2019. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r2.pdf.

PKCS#1 v2.2: RSA Cryptography Standard (RFC 8017), Internet Engineering Task Force (IETF), November 2016. https://tools.ietf.org/html/rfc8017.

Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, NIST Special Publication 800-38D, November 2007. Verfügbar unter http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf.

Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices, NIST Special Publication 800-38E, Januar 2010. Verfügbar unter https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38e.pdf.

Recommendation for Key Derivation Using Pseudorandom Functions, NIST Special Publication 800-108, Oktober 2009, Verfügbar unter https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-108.pdf.

Recommendation for Key Management - Part 1: General (Revision 5), NIST Special Publication 800-57A, Mai 2020, Verfügbar unter https://doi.org/10.6028/NIST.SP.800-57pt1r5.

Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised), NIST Special Publication 800-56A Revision 3, April 2018. Verfügbar unter https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf .

Empfehlung für die Generierung von Zufallszahlen mit deterministischen Zufallsbitgeneratoren, NIST Special Publication 800-90A Revision 1, Juni 2015, verfügbar unter https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf.

SEC 2: Recommended Elliptic Curve Domain Parameters, Standards for Efficient Cryptography Group, Version 2.0, 27 Januar 2010.

Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS), Brown, D., Turner, S., Internet Engineering Task Force, Juli 2010, http://tools.ietf.org/html/rfc5753/.

X9.62-2005: Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), American National Standards Institute, 2005.

Dokumentverlauf für AWS KMS Cryptographic Details

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation für AWS Key Management Service Cryptographic Details beschrieben. Darüber hinaus aktualisieren wir die Dokumentation regelmäßig, um auf das Feedback einzugehen, das Sie uns schicken.

Änderung	Beschreibung	Datum
Aktualisierter Inhalt	Details über die Durchführung der AWS KMS-`ReplicateKey` -Operation hinzugefügt.	28. Oktober 2021
Änderung der Dokumentation	Ersetzen des Begriffes Kundenhauptschlüssel (Customer Master Key, CMK) mit AWS KMS key und KMS-Schlüssel.	30. August 2021
Erstversion	Dieser Leitfaden wurde aus dem technischen Papier des KMS Cryptographic Details erstellt	30. Dezember 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.